

APPENDICES I-III

10043355 044400  
201710 555500

## Appendix I

```

/*      Copyright (c) 1991-1995 Duane DeSieno
/***** END TRAIN.C
*****/

void FindVariables()
{
short- x, n, i, k;
long nIn;
long NumPasses;
struct ddnet FAR *pnet;
float HHUGE *TrnData;
FILE *fLog;
FILE *fp;
FILE *fEnum;
    /* load the structures */
    dd_get_struct(NetNum, &pnet);
    /* load the root network parameters */
    sprintf (ParFileName, "%s.par", RootName);
    dd_read_parms (NetNum, ParFileName);
    sprintf (ParFileName, "%s.vsp", RootName);
    sprintf (TrnFileName, "%s.trn", RootName);
    sprintf (LogFileName, "%s.vsl", RootName);

    /* read the partameters for variable selection from .vsp file */
    fp = fopen (ParFileName, "r");
    if (fp == NULL) {
        printf ("could not open variable selection parameters file!
\n")
        return;
    }

    fLog = fopen(LogFileName, "a");

    /* setup initial list */
    for(x=0; x<MaxVars; x++) ImpVar[x] = EXCLUDE;
    nAvailVars = 0;

    /* nPartition = 5; */
    fgets (str, 256, fp);
    nPartition = (short) atoi (str);
    fprintf (fLog, "nPartitions = %d\n", nPartition);
    printf ("nPartitions = %d\n", nPartition);

    /* nConsensus = 10; */
    fgets (str, 256, fp);
    nConsensus = (short) atoi (str);
    fprintf (fLog, "nConsensus = %d\n", nConsensus);
    printf ("nConsensus = %d\n", nConsensus);

    /* nTop = 10; */
    fgets (str, 256, fp);
    nTop = (short) atoi (str);
    fprintf (fLog, "nTop = %d\n", nTop);
    printf ("nTop = %d\n", nTop);

    /* pnet->TrainSize = 510; */
    fgets (str, 256, fp);
    pnet->TrainSize = atol (str);
    fprintf (fLog, "TrainSize = %ld\n", pnet->TrainSize);

```

```

printf ("TrainSize = %ld\n",pnet->TrainSize);

/* pnet->Sigma[0] = (REAL)500; */
fgets(str, 256, fp);
pnet->Sigma[0] = (REAL)atoi(str);
fprintf (fLog, "report every %d passes\n", (int)pnet->Sigma[0]);
printf ("report every %d passes \n", (int)pnet->Sigma[0]);

/* NumPasses = 999L; */
fgets(str, 256, fp);
NumPasses = atol(str);
fprintf (fLog, "NumPasses = %ld\n",NumPasses);
printf ("NumPasses = %ld\n",NumPasses);

/* setup the ChiSq and SA lists */
nAvailVars = 0;
for(n=0; n<pnet->MaxPEs[0]; n++) {
    fgets(str, 256, fp);
    ChiSqList[n] = (short)atoi(str);
    /* add code for initial set of vars */
    if(ChiSqList[n] < 0) {
        ChiSqList[n] = -ChiSqList[n];
        ImpVar[ChiSqList[n] - 1] = NORMUSE;
    }
    SAList[n] = (short) atoi(strchr(str, ',') + 1);
    /* add code to never use these vars */
    if (SAList[n] < 0) {
        SAList[n] = SAList[n];
        ImpVar[SAList[n] - 1] = NEVER;
    } else {
        nAvailVars += 1;
    }
    fprintf (fLog, "[%02d] ChiSq = %d SA = %d\n", n, ChiSqList[n], SAList[n]);
    printf ("%02d] ChiSq = %d SA = %d\n", n, ChiSqList[n], SAList[n]);
}

fprintf (fLog, "Available Variables = %d\n", n, AvailVars);

for(n=0; n<nConsensus; n++) {
    fgets(str, 256, fp);
    Seeds[n] = atol(str);
    fprintf (fLog, "[%02d] Seed = %ld\n", n, Seeds[n]);
    printf ("%02d] Seed = %ld\n", n, Seeds[n]);
}
fclose(fLog);
fclose(fp);

/* load in the training data */
MaxVars = pnet->MaxPEs[0];

ImpVarErr = (REAL)9999.0;
pnet->TestSize pnet->TrainSize / (long) nPartition;
pnet->Learn.Flag = 1;
dd_allocate_net(NetNum);

/* set up special processing for inputs */
dd_set_inputs_func(NetNum, partition_get_input_data);

if ( AllocTrn(NetNum, (short)1, (short) pnet->TrainSize+10) < 0) {

```

```

        printf ("Error Allocating Training set! \n");
        exit(0);
    }
    dd_get_trn_array(NetNum, &TrnData);

    ReadTrnSet (NetNum, (short)1, (short)pnet->TrainSize, TrnFileName);
    pnet->TrainSize -= pnet->TestSize;

    /* copy ImpVar list to InputFunction list */
    fLog = fopen(LogFileName, "a");
    nIn = 0;
    for(x = 0; x < MaxVars; x++) {
        if (ImpVar[x] == NORMUSE) {
            InputFunction[x] = NORMUSE;
            nIn++;
            printf ("1");
            fprintf (fLog, "1");
        } else if (ImpVar[x] == NEVER) {
            InputFunction[x] = EXCLUDE;
            printf(".");
            fprintf (fLog, ".");
        } else {
            InputFunction[x] = EXCLUDE;
            printf("0");
            fprintf(fLog, "0");
        }
    }

    printf(" initial selection \n");
    fprintf(fLog, " initial selection \n");
    fclose (fLog);

    if(nIn > 0) {
        /* train consensus of networks on the partitioned data */
        TrainSelection(0, nIn, NumPasses);
        ConsensusErr[0] /= (REAL)nConsensus;
        ConsensusClass[0] /= (REAL)nConsensus;
        printf("Initial Consensus Error %f Class %f \n",
            (float)ConsensusErr[0], (float)ConsensusClass[0]);
        fLog = fopen(LogFileName, "a");
        fprintf(fLog, "Initial Consensus Error %f Class %f \n",
            (float)ConsensusErr[0], (float)ConsensusClass[0]);
        fclose(fLog);
        ImpVarErr = ConsensusErr[0];
    }

    /* open enumeration file for reading */
    fEnum. = fopen ("Enum.lst", "r");
    if (fEnum != NULL) {
        while (fgets (str, 256, fEnum) != 0) {

            /* generate the combination from the string */
            x = 0;
            for (k = 0; k < MaxVars; k++) {
                if (str[k] == '0') {
                    InputFunction[k] = EXCLUDE;
                    printf("0");
                } else if (str[k] == '1') {
                    InputFunction[k] = NORMUSE;
                    printf("1");
                }
            }
        }
    }

```

```

        x++;
    } else {
        InputFunction[k] = EXCLUDE;
        printf ("?");
    }
}
printf ("n");

/* evaluate the combination */
/* train consensus of networks on the partitioned data */
TrainSelection (0, (long) (x) NumPasses);

/* statistics */
ConsensusErr[0] /= (REAL) nConsensus;
ConsensusClass[0] /= (REAL) nConsensus;
fLog = fopen (LogFileName, "a");
for (i = 0; i < MaxVars; i++) {
    if (InputFunction[i] == NORMUSE) {
        printf ("%2d,", (int) i+1);
        fprintf (fLog, "%2d, (int) (i+1);
    }
}
printf ("Consensus Error %f Class %f \n",
        (float) ConsensusErr[0], (float)
ConsensusClass[0]);
fprintf (fLog, "Consensus Error %f Class %f \n",
        (float) ConsensusErr[0], (float) ConsensusClass
[0];
    fclose (fLog);
}

fclose (fEnum);
}

#ifdef NOT
    for (x = 1; x <= nAvailVars; x++) {
        /* generate x at a time combinations */
        /* initialize the array */
        for (i = 0; i < x; i++) {
            NewVar[i] = i;
        }

        /* iterate through the combinations */
        do {
            /* set up InputFunction[] from NewVar[] */
            n = 0;
            k = 0;
            for (i = 0; i < MaxVars; i++) {
                InputFunction[i] = NORMUSE; /* EXCLUDE; */
                if (ImpVar[i] == NEVER) {
                    InputFunction[i] = EXCLUDE;
                    continue;
                }
                if (k < x && NewVar[k] == n) {
                    InputFunction[i] = EXCLUDE; /* NORMUSE; */
                    k += 1;
                }
                n += 1;
            }
        }
    }
}

```

```

/* evaluate the combination */
/* train consensus of networks an the partitioned data */
TrainSelection(0, (long) (nAvailVars - x) NumPasses);

/* statistics */
ConsensusErr[0] /= (REAL)nConsensus;
ConsensusClass[0] /= (REAL) nConsensus;
fLog = fopen(LogFileName,"a");
for(i = 0; i < MaxVars; i++) {
    if (InputFunction[i] == NORMUSE) {
        printf ("%2d,", (int)(i+1));
        fprintf (fLog, "%2d,", (int) (i+1));
    }
}
printf ("Consensus Error %f Class %f \n",
        (float) ConsensusErr[0], (float) ConsensusClass [0]);
fprintf (fLog, "Consensus Error %f Class %f \n",
        (float) ConsensusErr[0], (float)ConsensusClass[0]);
fclose(fLog);

/* geneerate next selection */
for(i = x-1; i>=0; i--) {
    NewVar [i] ++;
    for(k = i+1; k < x; k++) {
        NewVar[k] = NewVar[k-1] + 1;
    }
    if(NewVar[x-1] < nAvailVars) {
        break;
    }
}

} while (NewVar[x-1] < nAvailVars);
}
#else

/* start the process of generating the important variables */
do {
    /* training data contains all variables */
    /* use special array for getting inputs to network */
    /* determine the variables to use in the current run */
    /* build list from ChiSq and SA */
    nNewVar = 0;
    for(x = 0; x < MaxVars; x++) {
        if (ImpVar [SAList [x] -1] == EXCLUDE) {
            NewVar[nNewVar] = SAList[x] - (short) 1;
            ImpVar[SAList[x] -1] = USED;
            nNewVar++;
        }
        if (ImpVar [ChiSqList [x] -1] == EXCLUDE) {
            NewVar[nNewVar] = ChiSqList[x] - (short) 1;
            ImpVar[ChiSqList[x] -1] = USED;
            nNewVar++;
        }
        if(nNewVar >= nTop) break;
    }
    /* work through the list of new variables */
    fLog = fopen(LagFileName,"a");
    for (n = 0; n < nNewVar; n++) {
        /* copy ImpVar list to InputFunction list */
        nIn = 0;

```

```

for(x = 0; x < MaxVars; x++) {
    if(ImpVar[x] == NORMUSE) {
        InputFunction[x] = NORMUSE;
        nIn++;
        printf ("1");
        f printf (fLog, "1");
    } else if(ImpVar[x] == NEVER) {
        InputFunction[x] = EXCLUDE;
        printf(".");
        fprintf (fLog, ".");
    } else {
        InputFunction[x] = EXCLUDE;
        printf ("0");
        fprintf(fLog, "0");
    }
}
InputFunction [NewVar[n]] = NORMUSE;
nIn++;

printf("...+ %d\n",NewVar[n]+1);
fprintf(fLog,"...+ %d\n",NewVar[n]+1);
fclose(fLog);

/* train consensus of networks on the partitioned data */
TrainSelection(n,nIn,NumPasses);
ConsensusErr[n] /= (REAL)nConsensus;
ConsensusClass[n] /= (REAL)nConsensus;
printf("Var %d Consensus Error %f Class %f \n",
(int)NewVar[n]+1,
(float)ConsensusErr[n], (float)ConsensusClass[n]);
f Log = f open (LogFileName, "a");
fprintf(fLog, "Var %d Consensus Error %f Class %f \n",
(int) NewVar[n] +1,
(float) ConsensusErr[n] , (float)
ConsensusClass[n]);
fclose(fLog);
}
/* Test of the list of variables is complete */
/* Find the best variable based an error */
BestErr = (REAL)999999.0;
BestVar = -1;
for(n=0; n< nNewVar; n++) {
    if (ConsensusErr[n] < BestErr) {
        BestErr = ConsensusErr[n];
        BestClass = ConsensusClass[n];
        BestVar = NewVar[n];
    }
}
/* Is there a variable that improved the ImpVar list Error */
/* Add the variable to the list of important variables */
if(BestErr < ImpVarErr) {
    ImpVar[BestVar] = NORMUSE;
    ImpVarErr = BestErr;
    printf ("Added %d to Imp Var List Error = %f Class =
%f\n",
(int) BestVar+1, (float)BestErr, (float)
BestClass);
fLog = fopen (LogFileName, "a");
fprintf (fLog, "Added %d to Imp Var List Error = %f Class
= %f\n",

```

```

        (int) BestVar+1, (float)BestErr, (float)
BestClass);
        fclose(fLog);
        for(x=0; x<MaxVars; x++) {
            /* cleanup from build of new variables list */
            if (ImpVar [x] == USED) ImpVar [x] = EXCLUDE;
        }

    /* if no improvement or no variables remaining, stop */
    } while(BestVar != -1 && nNewVar > 0);

    /* report the list of Important Variables and the Network Error */
    fLog = fopen. (LogFileName, "a");
    for(x=0; x<MaxVars; x++) {
        if (ImpVar [x] == NORMUSE) {
            printf ("USE [%d]\n", (int) x+1);
            fprintf (fLog, "USE [%d]\n", (int) x+1);
        }
    }

#ifdef
    fclose (fLog);
    dd_free_net (NetNum);
    if(TrnData != NULL) {
        FreeTrn (NetNum);
        TrnData = NULL;
    }
}
#endif

```

## Appendix II

Copyright (c) 1991-1995 Adeza Biomedical Corporation

FORM1.FRM - 1

' Neural Network, Function Declarations

```

Declare Function LoadNet% Lib "TKSDLL.DLL" (ByVal Net%, ByVal NetNameS)
Declare Function AllocNet% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function FreeNet% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function ReadWeights% Lib "TKSDLL.DLL" (ByVal Net%, ByVal NetNameS)
Declare Function LoadWeights% Lib "TKSDLL.DLL" (ByVal Net%, ByVal NetNameS)
Declare Function ReadParms% Lib "TKSDLL.DLL" (ByVal Net%, ByVal NetNameS)
Declare Function LoadParms% Lib "TKSDLL.DLL" (ByVal Net%, ByVal NetNameS)
Declare Function WriteWeights% Lib "TKSDLL.DLL" (ByVal. Net%, ByVal
Net.NameS)
Declare Function SaveWeights% Lib "TKSDLL.DLL" (ByVal Net%, ByVal NetNameS)
Declare Function WriteParms% Lib "TKSDLL.DLL," (ByVal. Net%, ByVal
NetNameS)
Declare Function SaveParms% Lib "TKSDLL.DLL" (ByVal. Net%, ByVal NetNameS)
Declare Function PutInput# Lib "TKSDLL.DLL" (ByVal. Net%, ByVal. nIn%,
pIn#)
Declare Function PutState# Lib "TKSDLL.DLL" (ByVal Net%, ByVal Layer%,
Byval nSt%, pSt#)
Declare Function PutOutput# Lib "TKSDLL.DLL" (ByVal Net%, ByVal nSt%, pSt#)
Declare Function PutTrn# Lib "TKSDLL.DLL" (ByVal Net%, ByVal nIn%, pIn#)
Declare Function PutWeight# Lib "TKSDLL.DLL" (ByVal Net%, ByVal Layer%,
ByVal pe%, ByVal nWt%, pWt#)
Declare Function PutParm# Lib "TKSDLL.DLL" (ByVal Net%, ByVal ParmNameS,
ByVal Layer%, pWt#)
Declare Function GetInput# Lib "TKSDLL.DLL: (ByVal Net%, ByVal. nIn%)
Declare Function GetState# Lib "TKSDLL.DLL" (ByVal Net%, ByVal Layer%,
ByVal nSt%)
Declare Function GetOutput# Lib "TKSDLL.DLL" (ByVal Net%, ByVal nSt%)
Declare Function GetWeight# Lib "TKSDLL.DLL" (ByVal Net%, ByVal Layer%,
ByVal pe%, ByVal nWt%)
Declare Function GetParm# Lib "TKSDLL.DLL" (ByVal Net%, ByVal ParmNameS,
ByVal Layer%)
Declare Function GetTrn# Lib "TKSDLL.DLL" (ByVal Net%, ByVal nIn%)
Declare Function GetNumInputs% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function GetNumOutputs% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function GetNumPEs% Lib "TKSDLL.DLL" (ByVal Net%, ByVal Layer%)
Declare Function GetNumLayers% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function InitializeWts% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function Train.Net% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function IterateNet% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function IsNetAvail% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function PutGrade% Lib "TKSDLL.DLL" (ByVal Net%, pGrade#)
Declare Function GetWtsGrade# Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function AdjustWts% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function GetBestWts% Lib "TKSDLL.DLL" (ByVal. Net%)
Declare Function AllocTrn% Lib "TKSDLL.DLL" (ByVal Net%, ByVal
InclDesired%, ByVal NumExamples%)
Declare Function FreeTrn% Lib "TKSDLL.DLL" (ByVal Net%)
Declare Function PutTrnData# Lib "TKSDLL.DLL" (ByVal Net%, ByVal
InclDesired% ByVal Example%, ByVal Offset%, pVal#)
Declare Function GetTrnData# Lib "TKSDLL.DLL" (ByVal Net%, ByVal
InclDesired%, ByVal Example%, ByVal Offset%)
Declare Function ReadTrnSet% Lib "TKSDLL.DLL" (ByVal Net%, ByVal
InclDesired%, ByVal NumExamples%, ByVal NetNameS)
Declare Function BatchTrain% Lib "TKSDLL.DLL" (ByVal Net%, ByVal MaxPasses%,
pTargetError#)

```

FORM1.FRM - 2

'Variables

Dim Age

Dim NetAge #

Dim NetPacks#

Dim NetBirth#

Dim NetPreg#

Dim NetAbort#

Dim NetDiabetes#

Dim NetPregHTN#

Dim NetHxEndo#

Dim NetDysmen#

Dim NetPelPain#

Dim NetPAP#

Dim NetHxPelSur#

Dim NetMedHx#

Dim NetGenWarts#

Dim NetElisa#

Sub RunNets ()

Con1 = 0

Con2 = 0

if NetElisa# = 0# Then

NetAge# = (Age - 32.07688) / 5.226876

For i = 0 To 7

a = PutInput (i, 1, NetAge#)

a = PutInput (i, 2, NetDiabetes#)

a = PutInput (i, 3, NetPregHTN#)

a = PutInput (i, 4, NetPacks#)

a = PutInput (i, 5, NetPreg#)

a = PutInput (i, 6, NetBirth#)

a = PutInput (i, 7, NetAbort#)

a = PutInput (i, 8, NetGenWarts#)

a = PutInput (i, 9, NetPAP#)

a = PutInput (i, 10, NetHxEndo#)

a = PutInput (i, 11, NetHxPelSur#)

a = PutInput (i, 12, NetMedHx#)

a = PutInput (i, 13, NetPelPain#)

a = PutInput (i, 14, NetDysmen#)

a = IterateNet (i)

Con1 = Con1 + GetState (i, 3, 1)

Con2 = Con2 + GetState (i, 3, 2)

Next i

Else

NetAge# = Age

For i = 8 To 15

a = PutInput (i, 1, NetAge#)

a = PutInput (i, 2, NetDiabetes#)

a = PutInput (i, 3, NetPregHTN#)

a = PutInput (i, 4, NetPacks#)

a = PutInput (i, 5, NetPreg#)

a = PutInput (i, 6, NetBirth#)

a = PutInput (i, 7, NetAbort#)

a = PutInput (i, 8, NetGenWarts#)

FORM1 FRM - 3

a = PutInput (i, 9, NetPAP#)

a = PutInput (i, 10, NetHxEndo#)

a = PutInput (i, 11, NetHxPelSur#)

a = PutInput (i, 12, NetMedHx#)

```

        a = PutInput(i, 13, NetPelPain#)
        a = PutInput(i, 14, NetDysmen#)
        a = PutInput(i, 15, NetElisa#)
        a = IterateNet(i)
        Con1 = Con1 + GetState (i, 3, 1)
        Con2 = Con2 + GetState (i, 3, 2)
    Next i
End If
Con1 = Con1 / 8
Con2 = Con2 / 8
Text2.Text = Con1
Text4.Text = Con2

' Generate Score
If NetElisa# = 0# Then
    Score = (Con1 - Con2) * 25
Else
    Score = (Con1 - Con2) * 18
End If
Text8.Text = Score
End Sub

Sub Check1_Click ()
    NetDiabetes# = 1# - NetDiabetes#
    RunNets
End Sub

Sub Check2_Click ()
    NetDysmen# = 1# - NetDysmen#
    RunNets
End Sub

Sub Check3_Click ()
    NetPAP# = 1# - NetPAP#
    RunNets
End Sub

Sub Check4_Click ()
    NetPelPain# = 1# - NetPelPain#
    RunNets
End Sub

Sub Check5_Click ()
    NetHxPelSur# = 1# - NetHxPelSur#
    RunNets
End Sub

Sub Check6_Click ()
    NetMedHx# = 1# - NetMedHx#
    RunNets

FORM1.FRM - 4

End Sub

Sub Check7_Click ()
    NetGenwarts# = 1# - NetGenWarts#
    RunNets
End Sub

```

```

Sub Check8_Click ()
    NetPregHTN# = 1# - NetPregHTN#
    RunNets
End Sub

Sub Check9_Click ()
    NetHxEndo# = 1# - NetHxEndo#
    RunNets
End Sub

Sub Command1_Click()
    Age = 30
    Text1.Text = Age
    NetAge# = (Age - 32.07688) / 5.226876
    NetPacks# = 0#
    Text3.Text = NetPacks#
    Text2.Text = "Not Run"
    Text4.Text = "Not Run"
    NetPreg# = 0#
    Text5.Text = NetPreg#
    NetBirth# = 0#
    Text6.Text = NetBirth#
    NetAbort# = 0#
    Text7.Text = NetAbort#
    NetElisa# = 0#
    Text7.Text = Net.Elisa#
    NetDiabetes# = 0#
    Check1.Value = 0
    NetPregHTN# = 0#
    Check8.Value = 0
    NetHxEndo# = 0#
    Check9.Value = 0
    NetDysmen# = 0#
    Check2.Value = 0
    NetPelPain# = 0#
    Check4.Value = 0
    NetPAP# = 0#
    Check3.Value = 0
    NetHxPelSur# = 0#
    Check5.Value = 0
    NetMedHx# = 0#
    Check6.Value = 0
    NetGenWarts# = 0#
    Check7.Value = 0

```

End Sub

FORM1.FRM - 5

```

Sub Command2_Click ()
    End
End Sub

```

```

Sub Form_Load ()
    a = LoadNet(0, "pat07_0")
    If a <> 0 Then GoTo mess
    a = LoadNet(1, "pat07_1")
    If a <> 1 Then GoTo mess

```

```

a = LoadNet(2, "pat07_2")
If a <> 2 Then GoTo mess
a = LoadNet(3, "pat07_3")
If a <> 3 Then GoTo mess
a = LoadNet(4, "pat07_4")
If a <> 4 Then GoTo mess
a = LoadNet(5, "pat07_5")
If a <> 5 Then GoTo mess
a = LoadNet(6, "pat07_6")
If a <> 6 Then GoTo mess
a = LoadNet(7, "pat07_7")
If a <> 7 Then GoTo mess
a = LoadNet(8, "crfel2_0")
If a <> 8 Then GoTo mess
a = LoadNet(9, "crfel2_1")
If a <> 9 Then GoTo mess
a = LoadNet(10, "crfel2_2")
If a <> 10 Then GoTo mess
a = LoadNet(11, "crfel2_3")
If a <> 11 Then GoTo mess
a = LoadNet(12, "crfel2_4")
If a <> 12 Then GoTo mess
a = LoadNet(13, "crfel2_5")
If a <> 13 Then GoTo mess
a = LoadNet(14, "crfel2_6")
If a <> 14 Then GoTo mess
a = LoadNet(15, "crfel2_7")
mess:
If a <> 15 Then Text4.Text = a + "No GOOD"
'initialize variables
Age = 30
Text1.Text = Age
NetAge# = (Age - 32.07688) / 5.226876
NetPacks# = 0#
Text3.Text = NetPacks#
Text2.Text = "Not Run"
Text4.Text = "Not Run"
NetPreg# = 0#
Text5.Text = NetPreg#
NetBirth# = 0#
Text6.Text = NetBirth#
NetAbort# = 0#
Text7.Text = NetAbort#
NetElisa# = 0#
Text 9.Text = NetElisa#

```

FORM1.FRM - 6

```

NetDiabetes# = 0#
NetPregHTN# = 0#
NetHxEndo# = 0#
NetDysmen# = 0#
NetPelPain# = 0#
NetPAP# = 0#
NetHxPelSur = 0#
NetMedHx# = 0#
NetGenWarts# = 0#
End Sub

Sub Text1_Change ()

```

```

        Age = Val(Text1.Text)
        RunNets
    End Sub

    Sub Text1_LostFocus ()
        RunNets
    End Sub

    Sub Text3_Change ()
        NetPacks# = Val(Text3.Text)
        RunNets
    End Sub

    Sub Text 3_LostFocus ()
        RunNets
    End Sub

    Sub Text5_Change ()
        NetPreg# = Val(Text5.Text)
        RunNets
    End Sub

    Sub Text5_LostFocus ()
        RunNets
    End Sub

    Sub Text6_Change ()
        NetBirth# Val (Text6.Text)
        RunNets
    End Sub

    Sub Text6_LostFocus ()
        RunNets
    End Sub

    Sub Text7_Change ()
        NetAbort# = Val (Text7.Text)
        RunNets
    End Sub

    Sub Text7_LostFocus ()
        RunNets
    End Sub

```

FORM1.FRM - 7

```

    Sub Text9_Change ()
        If Val(Text9.Text) <= 0# Then
            NetElisa# = 0#
        Else
            NetElisa# = Log(Val(Text9.Text))
        End If
        RunNets
    End Sub

    Sub Text9_LostFocus ()
        RunNets
    End Sub

```

## Appendix III

Copyright (c) 1991-1995 Adeza Biomedical Corporation

aa\_nets.h revised 7/1/95

Copyright (c) 1991-1995 Logical Designs Consulting Inc.

/\*This include file works for both DLL and DOS environments \*/

/\*The following define determines the floating point precision \*/

/\* Do not change it unless you intend to all source files \*/

#define USE-DOUBLES

#ifdef USE\_DOUBLES

#define REAL double

#define SIG\_LIMIT 44.0

#else

#define REAL float

#define SIG\_LIMIT 30.0

#endif

/\* The following prevents multiple inclusion of this header file \*/

#ifndef \_AA\_NETS\_H

#ifndef \_AA\_NETS\_H\_

/\* The following prevents C++ compiler from mangling names \*/

#ifdef \_cplusplus

extern "C" {

#endif /\* -cplusplus \*/

#ifdef \_WINDOWS

#include &lt;windows.h&gt;

#endif

#include &lt;stdio.h&gt;

#include &lt;stdlib.h&gt;

#include &lt;math.h&gt;

#include &lt;string.h&gt;

/\* Uncomment the following to enable user messages \*/

#define AA\_ENABLE\_USER\_MESSAGES

#ifdef \_WINDOWS

#ifdef \_WIN32

#define HUGE

#define EXPORT

#else

#define HUGE huge

#define EXPORT \_export

#endif

#else

typedef unsigned short HANDLE;

#define PASCAL

#ifdef MSC\_APPL

#include &lt;malloc.h&gt;

#include &lt;conio.h&gt;

#define HUGE huge

#define FAR far

#define EXPORT

#endif

```

#ifdef BC_APPL
#include <alloc.h>
#include <conio.h>
#define FAR
#define HUGE huge
#define EXPORT
#endif

#ifdef SC_APPL
#include <dos.h>
#include <conio.h>
#define FAR
#define HUGE huge
#define EXPORT
#endif

#ifdef UNIX_APPL
#define FAR
#define HUGE
#define EXPORT
#endif

#ifdef WD32_APPL
#include <conio.h>
#define FAR
#define HUGE
#define EXPORT
#endif

#endif

#define MAX_LAYERS 5
#define NU14-NETS 32

struct ddnet {
    /* Network Description Parameters */
    long NetArch; /* network interconnection arrangement */
    long nLayers; /* The total number of layers in the net */
    long MaxPEs[MAX_LAYERS] /* max Processing Elements (for Mallocs) */
    long nPEs[MAX_LAYERS]; /* number of hidden */
    long PEFunc[MAX_LAYERS]; /* Processing Element Function */
    long PETrans [MAX_LAYERS] /* Processing Element Transfer Function */
    long oIn(MAX_LAYERS); /* offset of Layer Inputs (init routine) */
    long oWts(MAX_LAYERS); /* offset of Weights (from init routine) */
    long oOut(MAX_LAYERS); /* Offset of Layer Outputs (init routine) */
    long nIn[MAX_LAYERS]; /* count of Layer Inputs (init routine) */
    long nWts[MAX_LAYERS]; /* total number of weights (init routine) */

    /* Network Training Parameters */
    long LearnFlag; /* 0=disable 1=enable */
    long BatchSize; /* parameter for batching */
};

```

```

long      TrainSize;      /* parameter for preprocessing */
long      TestSize;      /* parameter for preprocessing */
long      InitWtsFlag;    /* 0=No 1=Initialize weights */
long      RandSeed;      /* for random number generator */
long      NetErrorType;   /* kind of error to minimize by net
*/
REAL      ErrorTol;      /* Error Tolerance for training */
REAL      InputNoise;    /* Error Tolerance for training */
long      nTrialPES;     /* for growing algorithm, number of
trial units */
long      RcrOpsPerIter;  /* ops per iteration for recurrent
nets */
long      TrnSequence;    /* order of presentation of training
set */
long      TestWhileTrn;   /* Controls processing fro training
and testing */
long      ClassMethod;    /* Method used to to classification
performance measurement
*/
long      NetRule[MAX_LAYERS]; /* weight adjustment by
layer */
long      IterLimit[MAX_X LAYERS]; /* for growing algorithms
only */
REAL      InitWtsVal[MAX_LAYERS]; /* multiplier for grand ( ) */
REAL      XferOfs[MAX_LAYERS]; /* offset for XferPrime
*/
REAL      PESigma(MAX_LAYERS); /* Initial Sigma for L1, L2, RBF
*/
REAL      PEMu[MAX_LAYERS]; /* Learning factor for PESigma
*/
REAL      Alpha[MAX-LAYERS]; /* learning rule
parameters */
REAL      Beta[MAX_LAYERS]; /* Values dependent on learning
rule used */
REAL      Gamma [MAX_LAYERS];
REAL      Delta [MAX_LAYERS];
REAL      Epsilon [MAX_LAYERS];
REAL      Theta [MAX-YERS];
REAL      Lambda [MAX_LAYERS];
REAL      Mu [MAX_LAYERS];
REAL      Sigma [MAX_LAYERS];
REAL      WtsDecay (MAX_LAYERS);

/* Network pointers (not all are allocated for a given network) */
REAL      HUGE *pCurWts; /* pointer to current wieghts */
REAL      HUGE *pBestWts; /* pointer to best wieghts */
REAL      HUGE *pGateWts; /* pointer to spare weights */
REAL      HUGE *pDirWts; /* pointer to direction wieghts
*/
REAL      HUGE *pBiasWts; /* pointer to bias weights */
REAL      HUGE *pTempWts; /* pointer to spare weights */
REAL      HUGE *pNetSts; /* pointer to the weighted sums
states */
REAL      HUGE *pASts; /* pointer to the states for PE
outputs */
REAL      HUGE *pBSts; /* pointer to the states for PE
outputs */
REAL      HUGE *pDelSts; /* pointer to the states deltas
*/
REAL      HUGE *pTrnSts; /* pointer to the training states
*/

```

```

REAL      HUGE *pErrSts;      /* pointer to the Error stats */
REAL      HUGE *pPriorErrSts; /* pointer to the Prior Error
stats */
REAL      HUGE *pErrSumSts;   /* pointer to the Error Sum stats
*/
REAL      HUGE *pBiasSts;     /* pointer to the Bias stats */
REAL      HUGE *pProbSts;     /* pointer to the Prop stats */
REAL      HUGE *pCovMat;      /* pointer to covariance by
output & trial unit */
REAL      HUGE *pLastCovMat;  /* pointer to prior cov by output
& trial unit */
long      HUGE *poWts;        /* pointer to weights offsets by
pe element */

/* The following is to insure DLL compatibility */
HANDLE    hCurWts; /* HANDLE to current wieghts */
HANDLE    hBestWts; /* HANDLE to best wieghts */
HANDLE    hGateWts; /* HANDLE to spare weights */
HANDLE    hDirWts;  /* HANDLE to direction wieghts */
HANDLE    hBiasWts; /* HANDLE to bias weights */
HANDLE    hTempWts; /* HANDLE to spare weights */
HANDLE    hNetSts;  /* HANDLE to the weighted sums states */
HANDLE    hASts;    /* HANDLE to the states for PE outputs
*/
HANDLE    hBSts;    /* HANDLE to the states for PE outputs
*/
HANDLE    hDelSts;  /* HANDLE to the states deltas */
HANDLE    hTrnSts;  /* HANDLE to the training states */
HANDLE    hErrSts;  /* HANDLE to the Error stats */
HANDLE    hPriorErrSts; /* HANDLE to the Prior Error stats */
HANDLE    hErrSumSts; /* HANDLE to the Error Sum stats */
HANDLE    hBiasSts; /* HANDLE to the Bias stats */
HANDLE    hProbSts; /* HANDLE to the Prop stats */
HANDLE    hCovMat;  /* HANDLE to covariance by output & trial
unit */
HANDLE    hLastCovMat; /* HANDLE to prior cov by output & trial
unit */
HANDLE    hoWts;    /* HANDLE to weights offsets by pe element
*/

/* Network Training Statistics and Globals */
long      Iteration; /* iteration count */
long      OperMode;
long      TrialPick;
long      CurCnt[MAX-LAYERS];
long      TrainingMode; /* In Training Testing or
Sensitivity analysis */
long      TrnMaxErrSample; /* Training Example with
Maximum Error */
long      TrnClassCorrect; /* Training set Correct
count */
long      TstMaxErrSample; /* Test Example with
Maximum Error */
long      TstClassCorrect; /* Training set Correct
count */
REAL      TrnError; /* Training Set Error
Statistic */
REAL      TrnMaxError; /* Training Set Error
Statistic */
REAL      TrnClassPercent; /* Training Set Error
Statistic */

```

```

REAL      TstError;          /* Test Set Error Statistic
*/
REAL      TstMaxError;
Statistic */
REAL      TstClassPercent;   /* Test Set Error Statistic
*/
REAL      PETemp,MAX_LAYERSI; /* Temperature for Hopfield
MFA networks */
REAL      LastVal(MAX_LAYERS); /* current step size */
REAL      CurTemp[MAX_LAYERS];
REAL      CurErr[MAX_LAYERS]; /* to error function value
*/
REAL      LastErr(MAX_LAYERS); /* to error function value
*/
REAL      BestErr[MAX_LAYERS]; /* best error value */
};

```

```

#ifndef max
#define max(a,b) (((a)>(b))?(a):(b))
#define min(a,b) (((a)<(b))?(a):(b))
#endif

#ifndef fabs
#define fabs(a) (((a)>=0.0)?(a):(-a))
#endif

#ifndef ffsign
#define ffsign(a) (((a)>0.0)?(1.0):(((a)==0.0)?(0.0):(-1.0)))
#endif

```

```

/* DEFINES for input layer preprocessing */
#define NO_PREPROC 0
#define MEAN_STD 1
#define MAX_MIN 2
#define SUM_1 3
#define SUM_SQ-1 4

```

```

-/* DEFINES for Network Error form */
#define MEAN_SQ_ERR 1
#define MEAN_ABS_ERR 2
#define HYPER_SQ_ERR 3
#define BI_HYPER_SQ_ERR 4
#define MEAN_4PW_ERR 5
#define CROS_ENTROPY 6
#define CLASS_ERR 7
#define USER_DEFINED 8

```

```

/* DEFINES for Network Architecture */
#define FEED_FORWARD 1
#define FF_CON_PRIOR 2
#define TOTAL_RCR 3
#define PRIOR_RCR 4
#define CASCADE 5
#define CASCADE_RCR 6
#define ELMAN_RCR 7
#define JORDAN_RCR 8

```

```

/* DEFINES for the PE Functions */
#define DOT_PROD 1
#define L2_DIST 2
#define L1_DIST 3

```

```

#define QUAD_SUM 4
#define RADIAL 5
#define SIGMA_PI 6
#define GRNN_SUM 7
#ifdef AG_CUSTOM
#define FUZZ_APP 8
#define GEN_SIG_PI 9
#endif

/* DEFINES for Transfer Functions */
#define SIGMOID 1
#define BI_SIGMOID 2
#define ATAN 3
#define BI_ATAN 4
#define SIN 5
#define BI_SIN 6
#define LINEAR 7
#define THRES_LINEAR 8
#define BI_THRES_LINEAR 9
#define THRESHOLD 10
#define BI_THRESHOLD 11
#define GAUSS 12
#define CAUCHY 13
#define WIN_TAKE_ALL 14
#define PERIODIC_SIN 15
#define STCH_THRES 16
#define STCH_BI_THRES 17
#define MFA_THRES 18
#define MFA_BI_THRES 19

/* DEFINES for Training set ordering */
#define NORMAL 0
#define RANDOM 1
#define SHUFFLE 2
#define TD_REVERSE 3

/* DEFINES for Learning Rules for NetRule [layer] */
#define NONE 0
#define BACK_PROP 1
#define QUICK_PROP 2
#define JACOBS_PROP 3
#define KOHONEN_WTA 4
#define SIM_ANNEAL 5
#define RECURRENT_BP 6
#define KOHONEN_LVQ 7
#define CASCADE_CORR 8
#define SW_RAND_OPT 9
#define SIMPLEX_SA 10
#define POWELL_OPT 11
#define CONJ_GRAD 12
#define PROB_NET 13
#define GEN_REG_NET 14
#define LEVEN_MARQ 15
#define NUM_ALGO 16

/* DEFINES for CASCADE_CORR growing algorithms OperMode */
#define TRIAL_ADJ 1
#define OUTPUT_ADJ 2
#define GLOBAL_ADJ 3
#define MAX_CAPACITY 4

```

```

/* DEFINES for GRNN OperMode */
#define LOAD_TRN 1
#define SIGMA_ADJ 2

/* DEFINES for Classification Method */
#define BEST_PICK 0
#define WITHIN_TOL 1

/* The following is defined when error message displays should be shown */
#define AA_SHOW_ERROR_MESSAGES

/* Error return codes */
#define AA_ERROR_NONE 0
#define AA_ERROR_OPEN_PARDS_FILE -1
#define AA_ERROR_LOADING_PARDS -2
#define AA_ERROR_CREATE_PARDS_FILE -3
#define AA_ERROR_SAVING_PARDS -4
#define AA_ERROR_NO_EQUAL_IN_PARDS_LINE -5
#define AA_ERROR_IDENTIFIER_IN_PARDS -6
#define AA_ERROR_OPEN_WEIGHTS_FILE -7
#define AA_ERROR_LOADING_WEIGHTS -8
#define AA_ERROR_CREATE_WEIGHTS_FILE -9
#define AA_ERROR_SAVING_WEIGHTS -10
#define AA_ERROR_CREATE_WTS_LOG_FILE -11
#define AA_ERROR_SAVING_WTS_LOG -12

#define AA_ERROR_ALLOC -100
#define AA_ERROR_ALLOC_powts ( AA_ERROR_ALLOC - 0 )
#define AA_ERROR_ALLOC_pNetSts ( AA_ERROR_ALLOC - 1 )
#define AA_ERROR_ALLOC_pASts ( AA_ERROR_ALLOC - 2 )
#define AA_ERROR_ALLOC_pBSts ( AA_ERROR_ALLOC - 3 )
#define AA_ERROR_ALLOC_pDelSts ( AA_ERROR_ALLOC - 4 )
#define AA_ERROR_ALLOC_pTrnSts ( AA_ERROR_ALLOC - 5 )
#define AA_ERROR_ALLOC_pErrSts ( AA_ERROR_ALLOC - 6 )
#define AA_ERROR_ALLOC_pPriorErrSts ( AA_ERROR_ALLOC - 7 )
#define AA_ERROR_ALLOC_pErrSumSts ( AA_ERROR_ALLOC - 8 )
#define AA_ERROR_ALLOC_pBiasSts ( AA_ERROR_ALLOC - 9 )
#define AA_ERROR_ALLOC_pProbSts ( AA_ERROR_ALLOC - 10 )
#define AA_ERROR_ALLOC_pCovMat ( AA_ERROR_ALLOC - 11 )
#define AA_ERROR_ALLOC_pLastCovMat ( AA_ERROR_ALLOC - 12 )
#define AA_ERROR_ALLOC_pCurWts ( AA_ERROR_ALLOC - 13 )
#define AA_ERROR_ALLOC_pBestWts ( AA_ERROR_ALLOC - 14 )
#define AA_ERROR_ALLOC_pDirWts ( AA_ERROR_ALLOC - 15 )
#define AA_ERROR_ALLOC_pBiasWts ( AA_ERROR_ALLOC - 16 )
#define AA_ERROR_ALLOC_pGateWts ( AA_ERROR_ALLOC - 17 )
#define AA_ERROR_ALLOC_pTempWts ( AA_ERROR_ALLOC - 18 )

/* function prototypes reference */
/* Visual Basic and Excel functions specific to the DLL library */
short FAR PASCAL EXPORT LoadNet (short NetNum, char FAR *pName);
short FAR PASCAL EXPORT LoadWeights (short NetNum, char FAR *pName);
short FAR PASCAL EXPORT ReadWeights (short NetNum, char FAR *pName);
short FAR PASCAL EXPORT LoadParms (short NetNum, char FAR *pName);
short FAR PASCAL EXPORT ReadParms (short NetNum, char FAR *pName);
short FAR PASCAL EXPORT AllocNet (short NetNum);
short FAR PASCAL EXPORT FreeNet (short NetNum);
short FAR PASCAL EXPORT SaveWeights (short NetNum, char FAR *pName);
short FAR PASCAL EXPORT WriteWeights (short NetNum, char FAR *pName);
short FAR PASCAL EXPORT SaveParms (short NetNum, char FAR *pName);
short FAR PASCAL EXPORT WriteParms (short NetNum, char FAR *pName);

```

```

double FAR PASCAL EXPORT PutInput (short NetNum, short nIn, double FAR
*pIn );
double FAR PASCAL EXPORT PutState (short NetNum, short layer, short
pe, double FAR *pSt);
double FAR PASCAL EXPORT Putoutput (short NetNum, short nSt, double
FAR *pSt);
double FAR PASCAL EXPORT PutTrn (short NetNum, short nSt, double FAR
*pSt);
double FAR PASCAL EXPORT PutWeight (short NetNum, short layer, short
pe, short nWt, double FAR *pWt);
double FAR PASCAL EXPORT PutParm (short NetNum, char FAR *pName, short
layer, double FAR *pVal);
double FAR PASCAL EXPORT GetInput (short NetNum, short nIn);
double FAR PASCAL EXPORT GetState (short NetNum, short layer, short
nSt);
double FAR PASCAL EXPORT GetOutput (short NetNum, short nSt);
double FAR PASCAL EXPORT GetTrn (short NetNum, short nSt);
double FAR PASCAL EXPORT GetWeight (short NetNum, short layer, short
pe, short nWt);
double FAR PASCAL EXPORT GetParm (short NetNum, char FAR *pName, short
layer);
short FAR PASCAL EXPORT GetNumInputs (short NetNum);
short FAR PASCAL EXPORT GetNumOutputs (short NetNum);
short FAR PASCAL EXPORT GetNumPEs (short NetNum, short layer);
short FAR PASCAL EXPORT GetNumLayers (short NetNum);
short FAR PASCAL EXPORT InitializeWts (short NetNum);
short FAR PASCAL EXPORT TrainNet (short NetNum);
short FAR PASCAL EXPORT IterateNet (short NetNum);
short FAR PASCAL EXPORT IsNetAvail (short NetNum);
short FAR PASCAL EXPORT PutGrade (short NetNum, double FAR *pVal);
double FAR PASCAL EXPORT GetWtsGrade (short NetNum);
short FAR PASCAL EXPORT AdjustWts (short NetNum);
short FAR PASCAL EXPORT GetBestWts (short NetNum);
short FAR PASCAL EXPORT AllocTrn (short NetNum, short InclDesired, short
nExamples);
short FAR PASCAL EXPORT FreeTrn (short NetNum);
double FAR PASCAL EXPORT PutTrnData (short NetNum, short InclDesired,
short example, short offset, double FAR *pVal);
double FAR PASCAL EXPORT GetTrnData (short NetNum, short InclDesired,
short example, short offset);
short FAR PASCAL EXPORT ReadTrnSet (short NetNum, short InclDesired, short
MaxTrn, char FAR *pName );
short FAR PASCAL EXPORT BatchTrain (short NetNum, short MaxPasses, double
FAR *TargetError );

/* user definable network evaluation function for graded and batched
learning */
void FAR PASCAL EXPORT eval_net (short NetNum, REAL *pRMSError, REAL
*pMaxError, REAL *pC lassError);
void FAR PASCAL EXPORT dd_set_inputs_func (short NetNum, long (FAR PASCAL
EXPORT *inputs_fn) (short NetNum, long example));
void FAR PASCAL EXPORT dd_set_sample_func (short NetNum, void (FAR PASCAL
EXPORT *sample_fn) (short NetNum, long example));
void FAR PASCAL EXPORT dd_set_pass_func (short NetNum, void (FAR PASCAL
EXPORT *pass_fn) (short NetNum));

/* C language callable functions */
void FAR PASCAL dd_get_struct (short NetNum, struct ddnet FAR **pnet);
void FAR PASCAL dd_get_trn_array (short NetNum, float HUGE **ptrndata);
short FAR PASCAL dd_allocate_net (short NetNum);
void FAR PASCAL dd_initialize_wts (short NetNum);

```

```

void FAR PASCAL dd_free_net (short NetNum);
void FAR PASCAL dd_adjwts (short NetNum);
void FAR PASCAL dd_train_network (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_sa (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_swro (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_meb (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_pow (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_cg (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_pnn (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_grnn (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_lm (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train_by_sample (short NetNum, long MaxPasses, double
TargetError);
void FAR PASCAL dd_train (short NetNum);
void FAR PASCAL dd_iterate (short NetNum);
void FAR PASCAL dd_preproc (short NetNum);
void FAR PASCAL dd_gendir (short NetNum, short layer);
void FAR PASCAL dd_bstwts (short NetNum, short layer);
void FAR PASCAL dd_curwts (short NetNum, short layer);
void FAR PASCAL dd_otp_ff (short NetNum);
void FAR PASCAL dd_otp_ffcp (short NetNum);
void FAR PASCAL dd_otp_ti (short NetNum);
void FAR PASCAL dd_otp_pi (short NetNum);
void FAR PASCAL dd_otp_cas (short NetNum);
void FAR PASCAL dd_otp_cas_rcr (short NetNum);
void FAR PASCAL dd_otp_elm_rcr (short NetNum);
void FAR PASCAL dd_otp_jor_rcr (short NetNum);
void FAR PASCAL dd_grad (short NetNum);
void FAR PASCAL dd_grad_mse (short NetNum, short layer);
void FAR PASCAL dd_grad_mae (short NetNum, short layer);
void FAR PASCAL dd_grad_hse (short NetNum, short layer);
void FAR PASCAL dd_grad_bhse (short NetNum, short layer);
void FAR PASCAL dd_grad_m4pe (short NetNum, short layer);
void FAR PASCAL dd_grad_ce (short NetNum, short layer);
void FAR PASCAL dd_grad_y (short NetNum, short layer);
void FAR PASCAL dd_grad_ff (short NetNum, short layer);
void FAR PASCAL dd_grad_ffcp (short NetNum, short layer);
void FAR PASCAL dd_grad_t_rcr (short NetNum);
void FAR PASCAL dd_grad_cas (short NetNum, short layer);
void FAR PASCAL dd_grad_elm_rcr (short NetNum, short layer);
void FAR PASCAL dd_grad_jor_rcr (short NetNum, short layer);
void FAR PASCAL dd_adj_bpn (short NetNum, short layer);
void FAR PASCAL dd_adj_qp (short NetNum, short layer);
void FAR PASCAL dd_adj_jacob (short NetNum, short layer);
void FAR PASCAL dd_adj_koh (short NetNum, short layer);
void FAR PASCAL dd_adj_lvq (short NetNum, short layer);
void FAR PASCAL dd_adj_sa (short NetNum, short layer);
void FAR PASCAL dd_adj_swro (short NetNum, short layer);
void FAR PASCAL dd_grad_cascor (short NetNum);
void FAR PASCAL dd_adj_cascor (short NetNum);
void FAR PASCAL dd_adj_pnn (short NetNum);
void FAR PASCAL dd_adj_grnn (short NetNum);

```

```

void FAR PASCAL dd_adj_lm (short NetNum);
void FAR PASCAL dd_parms (short NetNum);
short FAR PASCAL dd_load_parms (short NetNum, char *name);
short FAR PASCAL dd_save_parms (short NetNum, char *name);
short FAR PASCAL dd_read_parms (short NetNum, char *name);
short FAR PASCAL dd_write_parms (short NetNum, char *name);
short FAR PASCAL dd_load_wts (short NetNum, char *name);
short FAR PASCAL dd_save_wts (short NetNum, char *name);
short FAR PASCAL dd_read_wts (short NetNum, char *name);
short FAR PASCAL dd_write_wts (short NetNum, char *name);
void FAR PASCAL dd_print_weights (short NetNum. );
short FAR PASCAL dd_log_weights (short NetNum, char *fname);
void FAR PASCAL dd_add_pe (short NetNum, long layer);
void FAR PASCAL generate_offsets (short NetNum, long *pTotWts, long
*pMaxWts);
void FAR PASCAL user_message (char *str );
char FAR *dd_getmem (HANDLE *pH, long len);
void FAR PASCAL dd_freemem (HANDLE *pH, char FAR *pM)
void FAR PASCAL add_wts (
    short NetNum,
    long layer,
    long ofs,
    long cnt,
    double InitVal);
void FAR PASCAL XferFunc(
    REAL HUGE *pIn,
    REAL HUGE *pOut,
    short n,
    short Type,
    REAL *Temp);
void FAR PASCAL XferPrime(
    REAL HUGE *pI,
    REAL HUGE *pN,
    REAL HUGE *pO,
    short n,
    short Type);
void FAR PASCAL PeFunc(
    REAL HUGE *pIn,
    REAL HUGE **ppWts,
    REAL HUGE *pOut,
    short nIn,
    short nOut,
    short Type);
void FAR PASCAL PePrime(
    REAL HUGE *pIn,
    REAL HUGE *pErrIn,
    REAL HUGE *pWts,
    REAL HUGE *pDir,
    REAL HUGE *pErrOut,
    REAL HUGE *pMu,
    short nIn,
    short nOut,
    short Type);
void FAR PASCAL vamul(
    REAL HUGE *pA,
    REAL HUGE *pvA,
    REAL HUGE *pB,
    REAL HUGE *pC,
    long n)
double FAR PASCAL crand (void);
double FAR PASCAL grand(void);

```

```

void FAR PASCAL surand (long idum);
double FAR PASCAL urand (void);
double FAR PASCAL xrand (void);

#ifdef _cplusplus
}
#endif

#endif /* AA_NETS_H_ */

/*****
*****/

// mainfrm.cpp : implementation of the CMainFrame class
//

#include "stdafx.h"
#include "PTDinp.h"

#include "mainfrm.h"

#ifdef _DEBUG
#undef THIS_FILE
static char BASED_CODE THIS_FILE[] = FILE;
#endif

////////////////////////////////////
// CMainFrame

IMPLEMENT_DYNCREATE (CMainFrame, CFrameWnd)

BEGIN_MESSAGE_MAP (CMainFrame, CFrameWnd)
    //{{AFX_MSG_MAP (CMainFrame)
        // NOTE - the ClassWizard will add and remove mapping macros
        here.
        // DO NOT EDIT what you see in these blocks of generated
        code !
        ON_WM_CREATE()
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

////////////////////////////////////
// arrays of IDs used to initialize control bars

// toolbar buttons - IDs are command buttons
static UINT BASED_CODE buttons[]
{
    // same order as in the bitmap 'toolbar.bmp'
    ID_FILE_OPEN,
    ID_SEPARATOR,
    ID_REC_FIRST,
    ID_REC_PREV,
    ID_REC_NEXT,
    ID_REC_LAST,
    ID_SEPARATOR,
    ID_DATA_EDIT,

```

```

        ID_DATA_NEW,
            ID_SEPARATOR,
        ID_REC_GOTO,
            ID_SEPARATOR,
        ID_APP_ABOUT,
    };

static UINT BASED_CODE indicators[] =
{
    ID_SEPARATOR,          // status line indicator
    ID_INDICATOR_CAPS,
    ID_INDICATOR_NUM,
    ID_INDICATOR_SCRL,
};

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction

CMainFrame::CMainFrame()
{
    // TODO: add member initialization code here
}

CMainFrame::~CMainFrame()
{
}

int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
        return -1;

    if (!m_wndToolBar.Create(this) ||
        !m_wndToolBar.LoadBitmap(IDR_MAINFRAME) ||
        !m_wndToolBar.SetButtons(buttons,
            sizeof(buttons)/sizeof(UINT)))
    {
        TRACE("Failed to create toolbar\n");
        return -1; // fail to create
    }

    if (!m_wndStatusBar.Create(this) ||
        !m_wndStatusBar.SetIndicators(indicators,
            sizeof(indicators)/sizeof(UINT)))
    {
        TRACE("Failed to create status bar\n");
        return -1; // fail to create
    }

    return 0;
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CMainFrame diagnostics

#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
    CFrameWnd::AssertValid();
}

```

```

}

void CMainFrame::Dump (CDumpContext& dc) const
{
    CFrameWnd::Dump(dc);
}

#endif //_DEBUG

////////////////////////////////////
////////////////////////////////////
// CMainFrame message handlers

// mainfrm.h : interface of the CMainFrame class
//
////////////////////////////////////
////////////////////////////////////

class CMainFrame : public CFrameWnd
{
protected: // create from serialization only
    CMainFrame();
    DECLARE_DYNCREATE(CMainFrame)

//Attributes
public:

// Operations
public:

// Implementation
public:
    virtual ~CMainFrame();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump (CDumpContext& dc) const;
#endif

protected: // control bar embedded members
    CStatusBar  m_wndStatusBar;
    CToolBar    m_wndToolBar;

// Generated message map functions
protected:
   //{{AFX_MSG(CMainFrame)
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
    // NOTE - the ClassWizard will add and remove member functions here.
    //      DO NOT EDIT what you see in these blocks of generated code!
   //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

////////////////////////////////////
////////////////////////////////////

```

PTDDLgl.cpp : Defines the class behaviors for the application.

```
#include "stdafx.h"
#include "'PTDinp.h'"
#include "'PTDDLgl.h'"
```

```
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = _FILE_;
#endif
```

```
////////////////////////////////////
////////////////////////////////////
// CPTDInp dialog
```

```
CPTDInp::CPTDInp(CWnd* pParent /*=NULL*/)
: CDialog(CPTDInp::IDD, pParent)
{
    //((AFX_DATA_INIT(CPTDInp)
    m_DATE_OF_BIRTH = "";
    m_NAME_F = "";
    m_NAME_MI = "";
    m_1_COMP = FALSE;
    m_2_COMP = FALSE;
    m_3_COMP = FALSE;
    m_4_COMP = FALSE;
    m_5_COMP = FALSE;
    m_6_COMP = FALSE;
    m_ACOG_N = FALSE;
    m_ACOG_Y = FALSE;
    m_Antibiotics = FALSE;
    m_AntiHyper = FALSE;
    m_CervCerclage = FALSE;
    m_CervFirm = FALSE;
    m_CervMod = FALSE;
    m_CervSoft = FALSE;
    m_Corticosteroids = FALSE;
    m_Dililation1_2 = FALSE;
    m_Dililation2 = FALSE;
    m_Dililation2_3 = FALSE;
    m_Dililation3 = FALSE;
    m_DililationGt3 = FALSE;
    m_Dililation1 = FALSE;
    m_DililationLtl = FALSE;
    m_DililationUkn = FALSE;
    m_EGAatSample = "";
    m_EGAbyLMP = "";
    m_EGAbySONO = "";
    m_EthnicOriginAsian = FALSE;
    m_EthnicOriginBlack = FALSE;
    m_EthnicOriginHispanic = FALSE;
    m_EthnicOriginNativeAmerican = FALSE;
    m_EthnicOriginOther = FALSE;
    m_EthnicOriginWhite = FALSE;
    m_FFN_Neg = FALSE;
    m_FFN_Pos = FALSE;
    m_GestationalDiabetes = FALSE;
    m_HypertensiveDisorders = FALSE;
```

```

m_Insulin = FALSE;
m_LadID = "";
m_MedicationNone = FALSE;
m_ModicationUnknown = FALSE;
m_MultipleGestationQuads = FALSE;
m_MultipleGestationTriplets = FALSE;
m_MultipleGestationTwins = FALSE;
m_MaritalStatusDivorced = FALSE;
m_MaritalStatusLWP = FALSE;
m_MaritalStatusMarried = FALSE;
m_MaritalStatusOther = FALSE;
m_MaritalStatusSingle = FALSE;
m_MaritalStatusWidowed = FALSE;
m_MultipleGestation = FALSE;
m_PatientComp1 = FALSE;
m_PatientComp2 = FALSE;
m_PatientComp3 = FALSE;
m_PatientComp4 = FALSE;
m_PatientComp5 = FALSE;
m_PatientComp6 = FALSE;
m_Tocolytics = FALSE;
m_UtCervAbnormal = FALSE;
m_VaginalBleeding = FALSE;
m_VaginalBleedingGross = FALSE;
m_VaginalBleedingMed = FALSE;
m_VaginalBleedingTrace = FALSE;
m_2_COMP_1 = FALSE;
m_2_COMP_2 = FALSE;
m_2_COMP_3 = FALSE;
m_ABORTIONS = "";
m_PARITY = "";
m_PatComp1_1_3 = FALSE;
m_PatComp1_10_12 = FALSE;
m_PatComp1_4_6 = FALSE;
m_PatComp1_7_9 = FALSE;
m_PatComp1_GT12 = FALSE;
m_PatComp1_LT1 = FALSE;
m_GRAVITY = "";
//}}AFX_DATA_INTT
}

void CPTDInp::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CPTDInp)
    DDX_Text(pDX, IDC_DATE_OF_BIRTH, m_DATE_OF_BIRTH);
    DDX_Text(pDX, IDC_NAME_F, m_NAME_F);
    DDV_MaxChars(pDX, m_NAME_F, 24);
    DDX_Text(pDX, IDC_NAME_L, m_NAME_L);
    DDV_MaxChars(pDX, m_NAME_L, 24);
    DDX_Text(pDX, IDC_NAME_MI, m_NAME_MI);
    DDV_MaxChars(pDX, m_NAME_MI, 2);
    DDX_Check(pDX, IDC_1_COMP, m_1_COMP);
    DDX_Check(pDX, IDC_2_COMP, m_2_COMP);
    DDX_Check(pDX, IDC_3_COMP, m_3_COMP);
    DDX_Check(pDX, IDC_4_COMP, m_4_COMP);
    DDX_Check(pDX, IDC_5_COMP, m_5_COMP);
    DDX_Check(pDX, IDC_6_COMP, m_6_COMP);
    DDX_Check(pDX, IDC_ACOG_N, m_ACOG_N);
    DDX_Check(pDX, IDC_ACOG_Y, m_ACOG_Y);
    DDX_Check(pDX, IDC_ANTIBIOTICS, m_Antibiotics);
    //}}AFX_DATA_MAP
}

```

```

DDX_Check(pDX, IDC_ANTIHYPER, m_AntiHyper);
DDX_Check(pDX, IDC_CERV_CERCLAGE, m_CervCerclage);
DDX_Check(pDX, IDC_CERV_FIRM, m_CervFirm);
DDX_Check(pDX, IDC_CERV_MOD, m_CervMod);
DDX_Check(pDX, IDC_CERV_SOFT, m_CervSoft);
DDX_Check(pDX, IDC_CORTICOSTEROIDS, m_Corticosteroids);
DDX_Check(pDX, IDC_DILITATION_1_2, m_Dililation1_2);
DDX_Check(pDX, IDC_DILITATION_2, m_Dililation2);
DDX_Check(pDX, IDC_DILITATION_2_3, m_Dililation2_3);
DDX_Check(pDX, IDC_DILITATION_3, m_Dililation3);
DDX_Check(pDX, IDC_DILITATION_GT3, m_DililationGt3);
DDX_Check(pDX, IDC_DILITATION_1, m_Dililation1);
DDX_Check(pDX, IDC_DILITATION_LT1, m_DililationLt1);
DDX_Check(pDX, IDC_DILITATION_UKN, m_DililationUkn);
DDX_Text(pDX, IDC_EGA_AT_SAMP, m_EGAatSample);
DDV_MaxChars(pDX, m_EGAatSample, 10);
DDX_Text(pDX, IDC_EGA_BY_LMP, m_EGAbyLMP);
DDX_Text(pDX, IDC_EGA_BY_SONO, m_EGAbySONO);
DDX_Check(pDX, IDC_EO_ASIAN, m_EthnicOriginAsian);
DDX_Check(pDX, IDC_EO_BLACK, m_EthnicOriginBlack);
DDX_Check(pDX, IDC_EO_HISPANIC, m_EthnicOriginHispanic);
DDX_Check(pDX, IDC_EO_NATIVE_AMERICAN, m_EthnicoriginNativeAmerican);
DDX_Check(pDX, IDC_EO_OTHER, m_EthnicOriginOther);
DDX_Check(pDX, IDC_EO_WHITE, m_EthnicOriginWhite);
DDX_Check(pDX, IDC_FFN_NEG, m_FFN_Neg);
DDX_Check(pDX, IDC_FFN_POS, m_FFN_Pos);
DDX_Check(pDX, IDC_GEST_DIABETES, m_GestationalDiabetes);
DDX_Check(pDX, IDC_HYPERTEN_DISORDERS, m_HypertensiveDisorders);
DDX_Check(pDX, IDC_INSULIN, m_Insulin);
DDX_Text(pDX, IDC_LAB_ID, m_LadID);
DDX_Check(pDX, IDC_MED_NONE, m_MedicationNone);
DDX_Check(pDX, IDC_MED_UKN, m_MedicationUnknown);
DDX_Check(pDX, IDC_MG_QUADS, m_MultipleGestationQuads);
DDX_Check(pDX, IDC_MG_TRIPLETS, m_MultipleGestationTriplets);
DDX_Check(pDX, IDC_MG_TWINS, m_MultipleGestationTwins);
DDX_Check(pDX, IDC_MS_DIVORCED, m_MaritalStatusDivorced);
DDX_Check(pDX, IDC_MS_LWP, m_MaritalStatusLWP);
DDX_Check(pDX, IDC_MS_MARRIED, m_MaritalStatusMarried);
DDX_Check(pDX, IDC_MS_OTHER, m_MaritalStatusOther);
DDX_Check(pDX, IDC_MS_SINGLE, m_MaritalStatusSingle);
DDX_Check(pDX, IDC_MS_WIDOWED, m_MaritalStatusWidowed);
DDX_Check(pDX, IDC_MULT_GEST, m_MultipleGestation);
DDX_Check(pDX, IDC_PATIENT_COMP_1, m_PatientComp1);
DDX_Check(pDX, IDC_PATIENT_COMP_2, m_PatientComp2);
DDX_Check(pDX, IDC_PATIENT_COMP_3, m_PatientComp3);
DDX_Check(pDX, IDC_PATIENT_COMP_4, m_PatientComp4);
DDX_Check(pDX, IDC_PATIENT_COMP_5, m_PatientComp5);
DDX_Check(pDX, IDC_PATIENT_COMP_6, m_PatientComp6);
DDX_Check(pDX, IDC_TOCOLYTICS, m_Tocolytics);
DDX_Check(pDX, IDC_UT_CWRV_ABNORM, m_UtCervAbnormal);
DDX_Check(pDX, IDC_VAGINAL_BLEEDING, m_VaginalBleeding);
DDX_Check(pDX, IDC_VB_GROSS, m_VaginalBleedingGross);
DDX_Check(pDX, IDC_VB_MED, m_VaginalBleedingMed);
DDX_Check(pDX, IDC_VB_TRACE, m_VaginalBleedingTrace);
DDX_Check(pDX, IDC_2_COMP_1, m_2_COMP_1);
DDX_Check(pDX, IDC_2_COMP_2, m_2_COMP_2);
DDX_Check(pDX, IDC_2_COMP_3, m_2_COMP_3);
DDX_Text(pDX, IDC_ABORTIONS, m_ABORTIONS);
DDV_MaxChars(pDX, m_ABORTIONS, 2);
DDX_Text(pDX, IDC_PARITY, m_PARITY);
DDV_MaxChars(pDX, m_PARITY, 2);

```

```

DDX_Check(pDX, IDC_PC1_1_3, m_PatCompl_1_3);
DDX_Check(pDX, IDC_PC1_10_12, m_PatCompl_10_12);
DDX_Check(pDX, IDC_PC1_4_6, m_PatCompl_4_6);
DDX_Check(pDX, IDC_PC1_7_9, m_PatCompl_7_9);
DDX_Check(pDX, IDC_PC1_GT12, m_PatCompl_GT12);
DDX_Check(pDX, IDC_PC1_LT1, m_PatCompl_LT1);
DDX_Text(pDX, IDC_GRAVIDITY, m_GRAVITY);
DDV_MaxChars(pDX, m_GRAVITY, 2);
//}}AFX_DATA_MAP
}

BEGIN MESSAGE MAP(CPTDInp, CDialog)
//{{AFX_MSG_MAP(CPTDInp)
ON_WM_RBUTT5NDOWN()
ON_BN_CLICKED(IDC_ACOG_N, OnAcogN)
ON_BN_CLICKED(IDC_ACOG_Y, OnAcogY)
ON_BN_CLICKED(IDC_FFN_NEG, OnFfnNeg)
ON_BN_CLICKED(IDC_FFN_POS, OnFfnPos)
ON_BN_CLICKED(IDC_MG_QUADS, OnMgQuads)
ON_BN_CLICKED(IDC_MG_TRIPLETS, OnMgTriplets)
ON_BN_CLICKED(IDC_MG_TWINS, OnMgTwins)
ON_BN_CLICKED(IDC_MULT_GEST, OnMultGest)
ON_BN_CLICKED(IDC_DILITATION_1, OnDililation1)
ON_BN_CLICKED(IDC_DILITATION_1_2, OnDililation2)
ON_BN_CLICKED(IDC_DILITATTON_2, OnDililation2)
ON_BN_CLICKED(IDC_DILITATION_2_3, OnDililation23)
ON_BN_CLICKED(IDC_DILITATION_3, OnDililation3)
ON_BN_CLICKED(IDC_DILITATION_GT3, OnDililationGt3)
ON_BN_CLICKED(IDC_DILITATION_LT1, OnDililationLt1)
ON_BN_CLICKED(IDC_DILITATION_UKN, OnDililationUkn)
ON_BN_CLICKED(IDC_CERV_FIRM, OnCervFirm)
ON_BN_CLICKED(IDC_CERV_MOD, OnCervMod)
ON_BN_CLICKED(IDC_CERV_SOFT, OnCervSoft)
ON_BN_CLICKED(IDC_VAGINAL_BLEEDING, OnVaginalBleeding)
ON_BN_CLICKED(IDC_VB_GROSS, OnVbGross)
ON_BN_CLICKED(IDC_VB_MED, OnVbMed)
ON_BN_CLICKED(IDC_VB_TRACE, OnVbTrace)
ON_BN_CLICKED(IDC_2_COMP, On2Comp)
ON_BN_CLICKED(IDC_2_COMP_1, On2Comp1)
ON_BN_CLICKED(IDC_2_COMP_2, On2Comp2)
ON_BN_CLICKED(IDC_2_COMP_3, On2Comp3)
ON_BN_CLICKED(IDC_PATIENT_COMP_1, OnPatientComp1)
ON_BN_CLICKED(IDC_PC1_1_3, OnPc113)
ON_BN_CLICKED(IDC_PC1_10_12, OnPc11012)
ON_BN_CLICKED(IDC_PC1_4_6, OnPc146)
ON_BN_CLICKED(IDC_PC1_7_9, OnPc179)
ON_BN_CLICKED(IDC_PC1_GT12, OnPc1Gt12)
ON_BN_CLICKED(IDC_PC1_LT1, OnPc1Lt1)
ON_BN_CLICKED(IDC_EO_ASIAN, OnEoAsian)
ON_BN_CLICKED(IDC_EO_BLACK, OnEoBlack)
ON_BN_CLICKED(IDC_EO_HISPANIC, OnEoHispanic)
ON_BN_CLICKED(IDC_EO_NATIVE_AMERICAN, OnEoNativeAmerican)
ON_BN_CLICKED(IDC_EO_OTHER, OnEoOther)
ON_BN_CLICKED(IDC_EO_WHITE, OnEoWhite)
ON_BN_CLICKED(IDC_MS_DIVORCED, OnMsDivorced)
ON_BN_CLICKED(IDC_MS_LWP, OnMsLwp)
ON_BN_CLICKED(IDC_MS_MARRIED, OnMsMarried)
ON_BN_CLICKED(IDC_MS_OTHER, OnMsOther)
ON_BN_CLICKED(IDC_MS_SINGLE, OnMsSingle)
ON_BN_CLICKED(IDC_MS_WIDOWED, OnMsWidowed)
//}}AFX_MSG_MAP

```

```

END_MESSAGE_MAP()

////////////////////////////////////
////////////////////////////////////
// CPTDInp message handlers

BOOL CPTDInp::OnInitDialog()
{
    CDialog::OnInitDialog();

    // TODO: Add extra initialization here
    //MoveWindow(0,-250,500,500);          // one way to handle large
dialogs

    return TRUE;                          // return TRUE    unless you set the focus to
a control
}

void CPTDInp::OnRButtonDown(UINT nFlags, CPoint point)
{
    // TODO: Add your message handler code here and/or call default CRect
rect;
    GetWindowRect(&rect);
    CRect Desk;
    GetDesktopWindow()->GetWindowRect (&Desk);
    //char str[256];
    //sprintf (str, "t %d l %d b %d r %d \n t %d l %d b %d r %d ",
    //    rect.top, rect.left, rect.bottom, rect.right,
    //    Desk.top, Desk.left, Desk.bottom, Desk.right);
    //AfxMessageBox(str);
    if(rect.top < 0 ) {
        rect.bottom = rect.bottom - rect.top;
        rect.top = 0;
        MoveWindow(rect);
    } else if (rect.bottom > Desk.bottom) {
        rect.top = Desk.bottom - 3 - (rect.bottom - rect.top);
        rect.bottom = Desk.bottom 3;
        MoveWindow(rect);
    }

    CDialog::OnRButtonDown(nFlags, point);
}

void CPTDInp::OnAcogN()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_ACOG_N) {
        m_ACCOG_Y = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnAcogY()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_ACOG_Y) (

```

```

        m_ACOG_N = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnFfnNeg()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_FFN_Neg) {
        m_FFN_Pos = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnFfnPos()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_FFN_Pos) {
        m_FFN_Neg = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp: : OnMgQuads()
{
    // get current values from dialog
    updateData(TRUE);

    if(m_multipleGestationQuads) {
        m_MultipleGestation = TRUE;
        m_MultipleGestationTwins = FALSE;
        m_MultipleGestationTriplets = FALSE;
    } else {
        if(m_MultipleGestationTwins == FALSE &&
           m_MultipleGestationTriplets == FALSE ) {
            m_MultipleGestation = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnLMgTriplets()
{
    // get current values from dialog
    UpdateData(TRUE);
}

```

```

        if(m_MultipleGestationTriplets) {
            m_MultipleGestation = TRUE;
            m_MultipleGestationQuads = FALSE;
            m_MultipleGestationTwins = FALSE;
        } else {
            if ( m_MultipleGestationQuads == FALSE &&
                m_MultipleGestationTwins == FALSE ) {
                m_MultipleGestation = FALSE;
            }
        }

        // update dialog with new data
        UpdateData(FALSE);
    }

void CPTDInp::OriMgTwins()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_MultipleGestationTwins) {
        m_MultipleGestation = TRUE;
        m_MultipleGestationQuads = FALSE;
        m_MultipleGestationTriplets = FALSE;
    } else {
        if( m_MultipleGestationQuads == FALSE &&
            m_MultipleGestationTriplets == FALSE ) {
            m_MultipleGestation = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnMultGest()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_MultipleGestation) {
    }else {
        if(((CPTDInpApp*)AfxGetApp())->ClearSubfields) {
            m_MultipleGestationQuads = FALSE;
            m_MultipleGestationTriplets = FALSE;
            m_MultipleGestationTwins = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnDililation1()
{
    // get current values from dialog
    UpdateData(TRUE);

```

```

        if(m_Dilitation1) {
            m_Dilitation1_2 = FALSE;
            m_Dilitation2 = FALSE;
            m_Dilitation2_3 = FALSE;
            m_Dilitation3 = FALSE;
            m_DilitationGt3 = FALSE;
            //m_Dilitation1 = FALSE;
            m_DilitationLt1 = FALSE;
            m_DilitationUkn = FALSE;
        }

        // update dialog with new data
        UpdateData(FALSE);
    }

void CPTDInp::OnDilitation12()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_Dilitation1_2) {
        //m_Dilitation1_2 = FALSE;
        m_Dilitation2 = FALSE;
        m_Dilitation2_3 = FALSE;
        m_Dilitation3 = FALSE;
        m_DilitationGt3 = FALSE;
        m_Dilitation1 = FALSE;
        m_DilitationLt1 = FALSE;
        m_DilitationUkn = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnDilitation2()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_Dilitation2)
        m_Dilitation1_2 = FALSE;
        //m_Dilitation2 = FALSE;
        m_Dilitation2_3 = FALSE;
        m_Dilitation3 = FALSE;
        m_DilitationGt3 = FALSE;
        m_Dilitation1 = FALSE;
        m_DilitationLt1 = FALSE;
        m_DilitationUkn = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnDilitation23()
{

```

```

// get current values from dialog
UpdateData(TRUE);

if(m_Dilitation2_3) {
    m_Dilitation1_2 = FALSE;
    m_Dilitation2 = FALSE;
    //m_Dilitation2_3 = FALSE;
    m_Dilitation3 = FALSE;
    m_DilitationGt3 = FALSE;
    m_Dilitation1 = FALSE;
    m_DilitationLt1 = FALSE;
    m_DilitationUkn = FALSE;
}

// update dialog with new data
UpdateData(FALSE);
}

void CPTDInp::OnDilitation3()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_Dilitation3) {
        m_Dilitation1_2 = FALSE;
        m_Dilitation2 = FALSE;
        m_Dilitation2_3 = FALSE;
        //m_Dilitation3 = FALSE;
        m_DilitationGt3 = FALSE;
        m_Dilitation1 = FALSE;
        m_DilitationLt1 = FALSE;
        m_DilitationUkn = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnDilitationGt3()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_DilitationGt3) {
        m_Dilitation1_2 = FALSE;
        m_Dilitation2 = FALSE;
        m_Dilitation2_3 = FALSE;
        m_Dilitation3 = FALSE;
        //m_DilitationGt3 = FALSE;
        m_Dilitation1 = FALSE;
        m_DilitationLt1 = FALSE;
        m_DilitationUkn = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

```

```

void CPTDInp::OnDililationLt1()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_DililationLt1) {
        m_Dililation1_2 = FALSE;
        m_Dililation2 = FALSE;
        m_Dililation2_3 = FALSE;
        m_Dililation3 = FALSE;
        m_DililationGt3 = FALSE;
        m_Dililation1 = FALSE;
        //m_DililationLt1 = FALSE;
        m_DililationUkn = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnDililationUkn()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_DililationUkn)
        m_Dililation1_2 = FALSE;
        m_Dililation2 = FALSE;
        m_Dililation2_3 = FALSE;
        m_Dililation3 = FALSE;
        m_DililationGt3 = FALSE;
        m_Dililation1 = FALSE;
        m_DililationLt 1= FALSE;
        //m_DililationUkn = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnCervFirm()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_CervFirm) {
        m_CervMod = FALSE;
        m_CervSoft = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnCervMod()
{
    // get current values from dialog
    UpdateData(TRUE);
}

```

```

        if(m_CervMod) {
            m_CervFirm = FALSE;
            m_CervSoft = FALSE;
        }

        // update dialog with new data
        UpdateData(FALSE);
    }

void CPTDInp::OnCervSoft()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_CervSoft) {
        m_CervMod = FALSE;
        m_CervFirm = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp: : OnVaginalBleeding()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_VaginalBleeding) {
    } else {
        if(((CPTDInpApp*)AfxGetApp())->ClearSubfields) {
            m_VaginalBleedingGross = FALSE;
            m_VaginalBleedingMed = FALSE;
            m_VaginalBleedingTrace = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnVbGross()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_VaginalBleedingGross) {
        m_VaginalBleeding = TRUE;
        m_VaginalBleedingMed = FALSE;
        m_VaginalBleedingTrace = FALSE;
    } else {
        if(m_VaginalBleedingMed == FALSE &&
            m_VaginalBleedingTrace == FALSE ) {
            m_VaginalBleeding = FALSE;
        }
    }

    // update dialog with new data

```

```

        UpdateData(FALSE);
    }

void CPTDInp::OnVbMed()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_VaginalBleedingMed) {
        m_VaginalBleedingGross = FALSE;
        m_VaginalBleeding = TRUE;
        m_VaginalBleedingTrace = FALSE;
    } else {
        if(m_VaginalBleedingGross == FALSE &&
            m_VaginalBleedingTrace == FALSE ) {
            m_VaginalBleeding = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnVbTrace()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_VaginalBleedingTrace) {
        m_VaginalBleedingGross = FALSE;
        m_VaginalBleedingMed = FALSE;
        m_VaginalBleeding = TRUE;
    } else {
        if(m_VaginalBleedingMed == FALSE
            m_VaginalBleedingGross == FALSE ) {
            m_VaginalBleeding = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::On2Comp()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_2_COMP) {
    } else {
        if(((CPTDInpApp*)AfxGetApp())->ClearSubfields) {
            m_2_COMP_1 = FALSE;
            m_2_COMP_2 = FALSE;
            m_2_COMP_3 = FALSE;
        }
    }

    // update dialog with new data

```

```

        UpdateData(FALSE);
    }

void CPTDInp::On2Comp1()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_2_COMP_1) {
        m_2_COMP = TRUE;
        m_2_COMP_2 = FALSE;
        m_2_COMP_3 = FALSE;
    } else {
        if(m_2_COMP == 2 - FALSE &&
           m_2_COMP_3 == FALSE ) {
            m_2_COMP = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::on2Comp2()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_2_COMP_2) {
        m_2_COMP = TRUE;
        m_2_COMP_1 = FALSE;
        m_2_COMP_3 = FALSE;
    } else {
        if(m_2_COMP_1 == FALSE &&
           m_2_COMP_3 == FALSE ) {
            m_2_COMP = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::on2Comp3()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_2_COMP_3) {
        m_2_COMP = TRUE;
        m_2_COMP_2 = FALSE;
        m_2_COMP_1 = FALSE;
    } else {
        if(m_2_COMP_2 == FALSE &&
           m_2_COMP_1 == FALSE ) {
            m_2_COMP = FALSE;
        }
    }
}

```

```

        // update dialog with new data
        UpdateData(FALSE);
    }

void CPTDInp::OnPatientCompl()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_PatientCompl) {
    } else {
        if(((CPTDInpApp*)AfxGetApp())->ClearSubfields) {
            m_PatCompl_LT1 = FALSE;
            m_PatCompl_1_3 = FALSE;
            m_PatCompl_4_6 = FALSE;
            m_PatCompl_7_9 = FALSE;
            m_PatCompl_10_12 = FALSE;
            m_PatCompl__GT12 = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnPc113()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_PatCompl_1_3) {
        m_PatCompl_LT1 = FALSE;
        m_PatientCompl = TRUE;
        m_PatCompl_4_6 = FALSE;
        m_PatCompl_7_9 = FALSE;
        m_PatCompl_10_12 = FALSE;
        m_PatCompl_GT12 = FALSE;
    } else {
        if(m_PatCompl_LT1 == FALSE &&
            m_PatCompl_1_3 == FALSE &&
            m_PatCompl_4_6 == FALSE &&
            m_PatCompl_7_9 == FALSE &&
            m_PatCompl_10_12 == FALSE
            m_PatCompl_GT12 == FALSE ) {
            m_PatientCompl = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnPc11012()
{
    // get current values from dialog
    UpdateData(TRUE);
}

```

```

if(m_PatCompl_10_12) {
    m_PatCompl_LT1 = FALSE;
    m_PatCompl_1_3 = FALSE;
    m_PatCompl_4_6 = FALSE;
    m_PatCompl_7_9 = FALSE;
    m_PatientCompl = TRUE;
    m_PatCompl_GT12 = FALSE;
} else {

if(m_PatCompl_LT1 == FALSE &&
    m_PatCompl_1_3 == FALSE &&
    m_PatCompl_4_6 == FALSE &&
    m_PatCompl_7_9 == FALSE &&
    m_PatCompl_10_12 == FALSE &&
    m_PatCompl_GT12 == FALSE ) {
    m_PatientCompl = FALSE;
}

// update dialog with new data
UpdateData(FALSE);
}

void CPTDInp::OnPc146()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_PatCompl_4_6) {
        m_PatCompl_LT1 = FALSE;
        m_PatCompl_1_3 = FALSE;
        m_PatientCompl = TRUE;
        m_PatCompl_7_9 = FALSE;
        m_PatCompl_10_12 = FALSE;
        m_PatCompl_GT12 = FALSE;
    } else {
        if(m_PatCompl_LT1 == FALSE &&
            m_PatCompl_1_3 == FALSE &&
            m_PatCompl_4_6 == FALSE &&
            m_PatCompl_7_9 == FALSE &&
            m_PatCompl_10_12 == FALSE &&
            m_PatCompl_GT12 == FALSE ) {
            m_PatientCompl = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnPc179()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_PatCompl_7_9) {
        m_PatCompl_LT1 = FALSE;
        m_PatCompl_1_3 = FALSE;
        m_PatCompl_4_6 = FALSE;

```

```

        m_PatientComp1 = TRUE;
        m_PatComp1_10_12 = FALSE;
        m_PatComp1_GT12 = FALSE;
    } else {
        if (m_PatComp1_LT1 == FALSE &&
            m_PatComp1_1_3 == FALSE &&
            m_PatComp1_4_6 == FALSE &&
            m_PatComp1_7_9 == FALSE &&
            m_PatComp1_10_12 == FALSE &&
            m_PatComp1_GT12 == FALSE ) {
            m_PatientComp1 = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnPclGt12()
{
    // get current values from dialog
    UpdateData(TRUE);

    if (m_PatComp1_GT12) {
        m_PatComp1_LT1 = FALSE;
        m_PatComp1_1_3 = FALSE;
        m_PatComp1_4_6 = FALSE;
        m_PatComp1_7_9 = FALSE;
        m_PatComp1_10_12 = FALSE;
        m_PatientComp1 = TRUE;
    } else {
        if (m_PatComp1_LT1 == FALSE &&
            m_PatComp1_1_3 == FALSE &&
            m_PatComp1_4_6 == FALSE &&
            m_PatComp1_7_9 == FALSE &&
            m_PatComp1_10_12 == FALSE &&
            m_PatComp1_GT12 == FALSE ) {
            m_PatientComp1 = FALSE;
        }
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnPclLt1()
{
    // get current values from dialog
    UpdateData(TRUE);

    if (m_PatComp1_LT1) {
        m_PatientComp1 = TRUE;
        m_PatComp1_1_3 = FALSE;
        m_PatComp1_4_6 = FALSE;
        m_PatComp1_7_9 = FALSE;
        m_PatComp1_10_12 = FALSE;
        m_PatComp1_GT12 = FALSE;
    } else {
        if (m_PatComp1_LT1 == FALSE &&

```

```

        m_PatCompl_1_3 == FALSE &&
        m_PatCompl_4_6 == FALSE &&
        m_PatCompl_7_9 == FALSE &&
        m_PatCompl_10_12 == FALSE &&
        m_PatCompl_GT12 == FALSE ) {
            m_PatientCompl = FALSE;
        }
    }
    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnEoAsian()
{
#ifdef NOT
    // get current values from dialog
    UpdateData(TRUE);

    if(m_EthnicOriginAsian) {
        //m_EthnicOriginAsian = FALSE;
        m_EthnicOriginBlack = FALSE;
        m_EthnicOriginHispanic = FALSE;
        m_EthnicOriginNativeAmerican = FALSE;
        m_EthnicOriginOther = FALSE;
        m_EthnicOriginWhite = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
#endif
}

void CPTDInp::OnEoBlack()
{
#ifdef NOT
    // get current values from dialog
    UpdateData(TRUE);

    if(m_EthnicOriginBlack) {
        m_EthnicOriginAsian = FALSE;
        //m_EthnicOriginBlack = FALSE;
        m_EthnicOriginHispanic = FALSE;
        m_EthnicOriginNativeAmerican = FALSE;
        m_EthnicOriginOther = FALSE;
        m_EthnicOriginWhite = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
#endif
}

void CPTDInp::OnEoHispanic()
{
#ifdef NOT
    // get current values from dialog
    UpdateData(TRUE);

    if(m_EthnicOriginHispanic) {
        m_EthnicOriginAsian = FALSE;

```

```

        m_EthnicOriginBlack = FALSE;
        //m_EthnicOriginHispanic = FALSE;
        m_EthnicOriginNativeAmerican = FALSE;
        m_EthnicOriginOther = FALSE;
        m_EthnicOriginWhite = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
#endif
}

void CPTDInp::OnEoNativeAmerican()
{
#ifdef NOT
    // get current values from dialog
    UpdateData(TRUE);

    if(m_EthnicOriginNativeAmerican) {
        m_EthnicOriginAsian = FALSE;
        m_EthnicOriginBlack = FALSE;
        m_EthnicOriginHispanic = FALSE;
        //m_EthnicOriginNativeAmerican = FALSE;
        m_EthnicOriginOther = FALSE;
        m_EthnicOriginWhite = FALSE;
    }

    // ate dialog with new data
    UpdateData(FALSE);
#endif
}

void CPTDInp::OnEoOther()
{
#ifdef NOT
    // get current values from dialog
    UpdateData(TRUE);

    if(m_EthnicOriginOther) {
        m_EthnicOriginAsian = FALSE;
        m_EthnicOriginBlack = FALSE;
        m_EthnicOriginHispanic = FALSE;
        m_EthnicOriginNativeAmerican = FALSE;
        //m_EthnicOriginOther = FALSE;
        m_EthnicOriginWhite = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
#endif
}

void CPTDInp::OnEowhite()
{
#ifdef NOT
    // get current values from dialog
    UpdateData(TRUE);

    if(m_EthnicOriginWhite) {
        m_EthnicOriginAsian = FALSE;
        m_EthnicOriginBlack = FALSE;
        m_EthnicOriginHispanic = FALSE;
        m_EthnicOriginNativeAmerican = FALSE;

```

```

        m_EthnicOriginOther = FALSE;
        //m_EthnicOriginWhite = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
#endif
}

void CPTDInp::OnLMsDivorced()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_MaritalStatusDivorced) {
        //m_MaritalStatusDivorced = FALSE;
        m_MaritalStatusLWP = FALSE;
        m_MaritalStatusMarried = FALSE;
        m_MaritalStatusOther = FALSE;
        m_MaritalStatusSingle = FALSE;
        m_MaritalStatusWidowed = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnMsLwp()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_MaritalStatusLWP) {
        m_MaritalStatusDivorced = FALSE;
        //m_MaritalStatusLWP = FALSE;
        m_MaritalStatusMarried = FALSE;
        m_MaritalStatusOther = FALSE;
        m_MaritalStatusSingle = FALSE;
        m_MaritalStatusWidowed = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnMsMarried()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_MaritalStatusMarried) {
        m_MaritalStatusDivorced = FALSE;
        m_MaritalStatusLWP = FALSE;
        //m_MaritalStatusMarried = FALSE;
        m_MaritalStatusOther = FALSE;
        m_MaritalStatusSingle = FALSE;
        m_MaritalStatusWidowed = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

```

```

}

void CPTDInp::OnMsOther()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_MaritalStatusOther) {
        m_MaritalStatusDivorced = FALSE;
        m_MaritalStatusLWP = FALSE;
        m_MaritalStatusMarried = FALSE;
        //m_MaritalStatusOther = FALSE;
        m_MaritalStatusSingle = FALSE;
        m_MaritalStatusWidowed = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnMsSingle()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_MaritalStatusSingle) {
        m_MaritalStatusDivorced = FALSE;
        m_MaritalStatusLWP = FALSE;
        m_MaritalStatusMarried = FALSE;
        m_MaritalStatusOther = FALSE;
        //m_MaritalStatusSingle = FALSE;
        m_MaritalStatusWidowed = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnMsWidowed()
{
    // get current values from dialog
    UpdateData(TRUE);

    if(m_MaritalStatusWidowed) {
        m_MaritalStatusDivorced = FALSE;
        m_MaritalStatusLWP = FALSE;
        m_MaritalStatusMarried = FALSE;
        m_MaritalStatusOther = FALSE;
        m_MaritalStatusSingle = FALSE;
        //m_MaritalStatusWidowed = FALSE;
    }

    // update dialog with new data
    UpdateData(FALSE);
}

void CPTDInp::OnOK()
{
    double val;
    char str[32];

```

```

char *ps;
int m, d, y;

    UpdateData(TRUE);

    // Check the Date of Birth field
    // parse the data from mm/dd/yyyy

    strcpy(str, m_DATE_OF_BIRTH);
    m = atoi(str);
    if( m < 1 || m > 12) {
        AfxMessageBox("Please enter date in mm/dd/yy format. Month out
of range");
        return;
    }
    ps = strchr(str, '/');
    if( ps == NULL ) {
        AfxMessageBox ("Please enter date in mm/dd/yy format.");
        return;
    } else {
        ps++;
        d = atoi(ps);
        if( d < 1 || d > 31 ) {
            AfxMessageBox ("Please enter date in mm/dd/yy format. Day
out of range");
            return;
        }
    }

    ps = strchr(ps, '/');
    if( ps == NULL ) {
        AfxMessageBox ("Please enter date in mm/dd/yy format.");
        return;
    } else {
        ps++;
        y = atoi(ps);
        if( y < 30 ) y += 2000;
        if( y < 99 ) y += 1900;
    }

    // Check all boxes that are used by the network
    if(
        m_EthnicOriginAsian == FALSE &&
        m_EthnicOriginBlack == FALSE &&
        m_EthnicOriginHispanic == FALSE &&
        m_EthnicOriginNativeAmerican == FALSE &&
        m_EthnicOriginOther == FALSE &&
        m_EthnicOriginWhite == FALSE
    ) {
        AfxMessageBox ("Please make selection for Ethnic Origin");
        return;
    }

    if(
        m_MaritalStatusDivorced == FALSE &&
        m_MaritalStatusLWP == FALSE &&
        m_MaritalStatusMarried == FALSE &&
        m_MaritalStatusOther == FALSE &&
        m_MaritalStatusSingle == FALSE &&
        m_MaritalStatusWidowed == FALSE
    ) {

```

```

        AfxMessageBox ("Please make selection for Marital Status");
        return;
    }

    if(
        m_CervFirm == FALSE &&
        m_CervMod == FALSE &&
        m_CervSoft == FALSE
    ) {
        // AfxMessageBox ("Please make selection for Cervical
Consistency");
        //return;
    }

    if(
        m_Dilitation1_2 == FALSE &&
        m_Dilitation2 == FALSE &&
        m_Dilitation2_3 == FALSE &&
        m_Dilitation3 == FALSE &&
        m_DilitationGt3 == FALSE &&
        m_Dilitation1 == FALSE &&
        m_DilitationLt1 == FALSE &&
        m_DilitationUkn == FALSE
    ) {
        AfxMessageBox ("Please make selection for Dilatation");
        return;
    }

    if(
        m_FFN_Neg == FALSE &&
        m_FFN_Pos == FALSE
    ) {
        AfxMessageBox ("Please make selection for fFN Result");
        return;
    }

    val = (double)atof(m_EGAbySONO);
    if( val == 0.0 ) {
        AfxMessageBox ("Please enter value for EGA by SONO");
        return;
    }
    if( val < 24.0 || val > 45.0 ) {
        AfxMessageBox ("Value for EGA by SONO must be between 24.0 and
45.0 weeks");
        return;
    }

    val = (double)atof(m_EGAbyLMP);
    if(val == 0.0 ) {
        AfxMessageBox ("Please enter value for EGA by LMP");
        return;
    }
    if( val < 24.0 || val > 45.0 ) {
        AfxMessageBox("Value for EGA by LMP must be between 24.0 and
45.0 weeks");
        return;
    }

    val = (double)atof(m_EGAatSample);
    if( val == 0.0 ) {
        AfxMessageBox ("Please enter value for EGA at Sample");
    }

```

```

        return;
    }
    if( val < 24.0 || val > 45.0 ) {
        AfxMessageBox ("Value for EGA at Sample must be between 24.0
and 45.0 weeks");
        return;
    }

    strcpy(str,m_GRAVITY);
    if(str[0] == 0 ) {
        AfxMessageBox ("Please enter value for Gravity");
        return;
    }

    strcpy(str,m_PARITY);
    if( str[0] == 0 ) {
        AfxMessageBox ("Please enter value for Parity");
        return;
    }

    strcpy(str,m_ABORTIONS);
    if( str[0] == 0 ) {
        AfxMessageBox ("Please enter value for Abortions");
        return;
    }

    if(m_2_COMP == TRUE &&
        m_2_COMP_1 == FALSE &&
        m_2_COMP_2 == FALSE &&
        m_2_COMP_3 == FALSE ) {
        AfxMessageBox ("Please make selection under History of Preterm
Delivery");
        return;
    }

    if(m_VaginalBleedingMed == FALSE &&
        m_VaginalBleedingGross == FALSE &&
        m_VaginalBleeding == TRUE &&
        m_VaginalBleedingTrace == FALSE ) {
        AfxMessageBox ("Please make selection under Vaginal Bleeding");
        return;
    }

    if(m_MultipleGestation == TRUE &&
        m_MultipleGestationQuads == FALSE &&
        m_MultipleGestationTriplets == FALSE &&
        m_MultipleGestationTwins == FALSE ) {
        AfxMessageBox ("Please make selection under Multiple
Gestation");
        return;
    }

    if(m_PatientCompl == TRUE &&
        m_PatCompl_LT1 == FALSE &&
        m_PatCompl_1_3 == FALSE &&
        m_PatCompl_4_6 == FALSE &&
        m_PatCompl_7_9 == FALSE &&
        m_PatCompl_10_12 == FALSE &&
        m_PatCompl_GT112 == FALSE ) {
        AfxMessageBox ("Please select Number/hr under Uterine
contractions");
    }

```

```

        return;
    }

    CDialog::OnOK();
}

```

```

// PTDDg11.h : header file
//

```

```

////////////////////////////////////
////////////////////////////////////
// CPTDInp dialog

```

```

class CPTDInp : public CDialog
{
// Construction
public:
    CPTDInp(CWnd* pParent = NULL); // standard constructor

// Dialog Data
    //{AFX_DATA(CPTDInp)
    enum { IDD = IDD_D_PTD_INP };
    CString      m_DATE_OF_BIRTH;
    CString      m_NAME_F;
    CString      m_NAME_L;
    CString      m_NAME_MI;
    BOOL         m_1_COMP;
    BOOL         m_2_COMP;
    BOOL         m_3_COMP;
    BOOL         m_4_COMP;
    BOOL         m_5_COMP;
    BOOL         m_6_COMP;
    BOOL         m_ACOG_N;
    BOOL         m_ACOG_Y;
    BOOL         m_Antibiotics;
    BOOL         m_AntiHyper;
    BOOL         m_CervCerclage;
    BOOL         m_CervFirm;
    BOOL         m_CervMod;
    BOOL         m_CervSoft;
    BOOL         m_Corticosteroids;
    BOOL         m_Dililation1_2;
    BOOL         m_Dililation2;
    BOOL         m_Dililation2_3;
    BOOL         m_Dililation3;
    BOOL         m_DililationGt3;
    BOOL         m_Dililation1;
    BOOL         m_DililationLt1;
    BOOL         m_DililationUkn;
    CString      m_EGAatSample;
    CString      m_EGAbyLMP;
    CString      m_EGAbySONO;
    BOOL         m_EthnicOriginAsian;
    BOOL         m_EthnicOriginBlack;
    BOOL         m_EthnicOriginHispanic;
    BOOL         m_EthnicOriginNativeAmerican;
    BOOL         m_EthnicOriginOther;
    }
}

```

```

BOOL        m_EthnicOriginWhite;
BOOL        m_FFN_Neg;
BOOL        m_FFN_Pos;
BOOL        m_GestationalDiabetes;
BOOL        m_HypertensiveDisorders;
BOOL        m_Insulin;
CString     m_LadID;
BOOL        m_MedicationNone;
BOOL        m_MedicationUnknown;
BOOL        m_MultipleGestationQuads;
BOOL        m_MultipleGestationTriplets;
BOOL        m_MultipleGestationTwins;
BOCL        m_MaritalStatusDivorced;
BOOL        m_MaritalStatusLWP;
BOOL        m_MaritalStatusMarried;
BOOL        m_MaritalStatusOther;
BOOL        m_MaritalStatusSingle;
BOOL        m_MaritalStatusWidowed;
BOOL        m_MultipleGestation;
BOOL        m_PatientComp1;
BOOL        m_PatientComp2;
BOOL        m_PatientComp3;
BOOL        m_PatientComp4;
BOOL        m_PatientComp5;
BOOL        m_PatientComp6;
BOOL        m_Tocolytics;
BOOL        m_UtCervAbnormal;
BOOL        m_VaginalBleeding;
BOOL        m_VaginalBleedingGross;
BOOL        m_VaginalBleedingMed;
BOOL        m_VaginalBleedingTrace;
BOOL        m_2_COMP_1;
BOOL        m_2_COMP_2;
BOOL        m_2_COMP_3;
CString     m_ABORTIONS;
CString     m_PARITY;
BOOL        m_PatComp1_1_3;
BOOL        m_PatComp1_10_12;
BOOL        m_PatComp1_4_6;
BOOL        m_PatComp1_7_9;
BOOL        m_PatComp1_GT12;
BOOL        m_PatComp1_LT1;
CString     m_GRAVITY;
//}}AFX_DATA

```

Implementation

protected:

```

virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support

// Generated message map functions
//{{AFX_MSG(CPTDInp)
virtual        BOOL        OnInitDialog();
afx_msg void OnRButtonDown(UINT nFlags, CPoint point);
afx_msg void OnAcogN();
afx_msg void OnAcogY();
afx_msg void OnFfnNeg();
afx_msg void OnFfnPos();
afx_msg void OnMgQuads();
afx_msg void OnMgTriplets();
afx_msg void OnMgTwins();
afx_msg void OnMultGest();

```

```

afx_msg void OnDililation1();
afx_msg void OnDililation12();
afx_msg void OnDililation2();
afx_msg void OnDililation23();
afx_msg void OnDililation3();
afx_msg void OnDililationGt3();
afx_msg void OnDililationLt1();
afx_msg void OnDililationUkn();
afx_msg void OnCervFirm();
afx_msg void OnCervMod();
afx_msg void OnCervSoft();
afx_msg void OnVaginalBleeding();
afx_msg void OnVbGross();
afx_msg void OnVbMed();
afx_msg void OnVbTrace();
afx_msg void On2Comp();
afx_msg void On2Comp1();
afx_msg void On2Comp2();
afx_msg void On2Comp3();
afx_msg void OnPatientComp1();
afx_msg void OnPcl113();
afx_msg void OnPcl11012();
afx_msg void OnPcl146();
afx_msg void OnPcl179();
afx_msg void OnPclGt12();
afx_msg void OnPclLt1();
virtual void OnOK();
afx_msg void OnEoAsian();
afx_msg void OnEoBlack();
afx_msg void OnEoHispanic();
afx_msg void OnEoNativeAmerican();
afx_msg void OnEoOther();
afx_msg void OnEoWhite();
afx_msg void OnMsDivorced();
afx_msg void OnMsLwp();
afx_msg void OnMsMarried();
afx_msg void OnMsOther();
afx_msg void OnMsSingle();
afx_msg void OnEoWidowed();
//}}AFX_MSG
DECLARE_MESSAGE_MAP()
};

```

```
// ptdgoto.cpp : implementation file
/
```

```

#include "stdafx.h"
#include "ptdinp.h"
#include "ptdgoto.h"

```

```

#ifdef _DEBUG
#undef THIS_FILE
static char BASED_CODE THIS_FILE[] = FILE;
#endif

```

```

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CPtdGoto dialog

CPtdGoto::CPtdGoto(CWnd* pParent /*=NULL*/)
    : CDialog(CPtdGoto::IDD, pParent)
{
   //{{AFX_DATA_INIT(CPtdGoto)
    m_IDStr = "";
    m_GotoMode = -1;
    m_RecNum = 0;
    //}}AFX_DATA_INIT
}

void CPtdGoto::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CPtdGoto)
    DDX_Text(pDX, IDC_E_GOTO_ID_NUM, m_IDStr);
    DDX_Radio(pDX, IDC_R_GOTO_SEL1, m_GotoMode);
    DDX_Text(pDX, IDC_E_GOTO_REC_NUM, m_RecNum);
    DDV_MinMaxLong(pDX, m_RecNum, 0, 100000);
    //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CPtdGoto, CDialog)
    //{{AFX_MSG_MAP(CPtdGoto)
    // NOTE: the ClassWizard will add message map macros here
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CPtdGoto message handlers

// ptdgoto.h : header file
//

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CPtdGoto dialog

class CPtdGoto : public CDialog
{
// Construction
public:
    CPtdGoto(CWnd* pParent = NULL); // standard constructor

// Dialog Data
    //{{AFX_DATA(CPtdGoto)
    enum { IDD = IDD_D_GOTO };
    CString m_IDStr;
    int m_GotoMode;
    long m_RecNum;
    //}}AFX_DATA

// Implementation
protected:
    virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support

```

```

// Generated message map functions
//{{AFX_MSG(CPTdGoto)
// NOTE: the ClassWizard will add member functions here
//}}AFX_MSG
DECLARE_MESSAGE_MAP()
}

// PTDidoc.cpp : implementation of the CPTDinpDoc class
//

#include "stdafx.h"
#include "PTDinp.h"

#include "PTDidoc.h"
#include "PTDGoto.h"
#include "aa_nets.h"

#ifdef _DEBUG
#undef THIS_FILE
static char BASED_CODE THIS_FILE[] = _FILE_;
#endif

////////////////////////////////////
////////////////////////////////////
// CPTDinpDoc

IMPLEMENT_DYNCREATE(CPTDinpDoc, CDocument)

BEGIN_MESSAGE_MAP(CPTDinpDoc, CDocument)
//{{AFX_MSG_MAP(CPTDinpDoc)
ON_COMMAND(ID_REC_FIRST, OnRecFirst)
ON_COMMAND(ID_REC_LAST, OnRecLast)
ON_COMMAND(ID_REC_NEXT, OnRecNext)
ON_COMMAND(ID_REC_PREV, OnRecPrev)
ON_COMMAND(ID_FILE_OPEN, OnFileOpen)
ON_COMMAND(ID_BLD_NET_FILE, OnBldNetFile)
ON_COMMAND(ID_REC_GOTO, OnRecGoto)
ON_COMMAND(ID_FILE_MRU_FILE1, OnFileMruFile1)
ON_COMMAND(ID_FILE_MRU_FILE2, OnFileMruFile2)
ON_COMMAND(ID_FILE_MRU_FILE3, OnFileMruFile3)
ON_COMMAND(ID_FILE_MRU_FILE4, OnFileMruFile4)
//}}AFX_MSG_MAP
END_MESSAGE_MAP()

////////////////////////////////////
////////////////////////////////////
// CPTDinpDoc construction/destruction

CPTDinpDoc::CPTDinpDoc()
{
    CurRecord = 0;
    NumRecords = 0;
    strcpy(PathName, "");
    IDStr = "";
    GotoMode = 0;

    InitializeRec();
    LoadNets();
}

```

```

        m_NetPos1 = 0.0;
        m_NetNeg1 = 0.0;
        m_NetPos2 = 0.0;
        m_NetNeg2 = 0.0;
        m_NetPos3 = 0.0;
        m_NetNeg3 = 0.0;
    }

CPTDinpDoc::~CPTDinpDoc()
{
    (CPTDinpApp*) AfxGetApp()->m_pDoc = NULL;

    FreeNets();
}

BOOL CPTDinpDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
        return FALSE;

    ((CPTDinpApp*)AfxGetApp())->m_pDoc = this;
    // TODO: add reinitialization code here
    // (SDI documents will reuse this document)

    return TRUE;
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CPTDinpDoc serialization

void CPTDinpDoc::Serialize(CArchive& ar)
{
    if (ar.IsStoring())
    {
        // TODO: add storing code here
    }
    else
    {
        // TODO: add loading code here
    }
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CPTDinpDoc diagnostics

#ifdef _DEBUG
void CPTDinpDoc::AssertValid() const
{
    CDocument::AssertValid();
}

void CPTDinpDoc::Dump (CDumpContext& dc) const
{
    CDocument::Dump(dc);
}
#endif // _DEBUG

```

```

////////////////////////////////////
////////////////////////////////////
// CPTDinpDoc commands

```

```

void CPTDinpDoc::OnRecFirst()
{

```

```

    CurRecord = 0;
    get_rec(Rec);

```

```

}

```

```

void CPTDinpDoc::OnRecLast()
{

```

```

    CurRecord = NumRecords - 1;
    get_rec(Rec);

```

```

}

```

```

void CPTDinpDoc::OnRecNext()
{

```

```

    CurRecord = min(CurRecord + 1, NumRecords - 1);
    get_rec(Rec);

```

```

}

```

```

void CPTDinpDoc::OnRecPrev()

```

```

    CurRecord = max(CurRecord - 1, 0);
    get_rec(Rec);

```

```

}

```

```

////////////////////////////////////
////////////////////////////////////

```

```

void CPTDinpDoc::get_rec( char* pRec )
{

```

```

    FILE *fp;
    char *stmp;

```

```

    fp = fopen(PathName,"rb");
    if(fp==NULL) {
    } else {
        f seek (fp, (long)((REC_LENGTH + 2L) *CurRecord) , SEEK_SET);
        f read (pRec, sizeof (char), (REC_LENGTH + 2L), fp);
        fclose(fp);
    }

```

```

    m_LAB_ID = get_fld(pRec,1,12);
    m_NAME_L = get_fld(pRec,13,24);
    m_NAME_F = get_fld(pRec,37,24);
    m_NAME_MI = get_fld(pRec,61,2);
    m_DATE_OF_DATA_ENTRY = get_fld(pRec, 63, 10) //time
    m_PATIENT_AGE = (double) atof (get_fld (pRec, 73, 20))
    m_DATE_OF_BIRTH = get_fld(pRec,93,10);
    //stmp = get_fld(pRec,103,2);
    //if(stmp[0] == '1') m_ETHNIC_ORIGIN_WHITE = ("1"); else
m_ETHNIC_ORIGIN_WHITE = ("0");
    //if(stmp[0] == '2') m_ETHNIC_ORIGIN_BLACK = ("1"); else
m_ETHNIC_ORIGIN_BLACK = ("0");

```

```

//if(stmp[0] == '3') m_ETHNIC_ORIGIN_ASIAN = ("1"); else
m_ETHNIC_ORIGIN_ASIAN = ("0");
//if(stmp[0] == '4') m_ETHNIC_ORIGIN_HISPANIC = ("1"); else
m_ETHNIC_ORIGIN_HISPANIC = ("0");
//if(stmp[0] == '5') m_ETHNIC_ORIGIN_NATIVE_AMERICAN = ("1"); else
m_ETHNIC_ORIGIN_NATIVE_AMERICAN = ("0");
//if(stmp[0] == '6') m_ETHNIC_ORIGIN_OTHER = ("1"); else
m_ETHNIC_ORIGIN_OTHER = ("0");
m_ETHNIC_ORIGIN_WHITE = get_fld(pRec,103,2);
m_ETHNIC_ORIGIN_BLACK = get_fld(pRec,105,2);
m_ETHNIC_ORIGIN_ASIAN = get_fld(pRec,107,2);
m_ETHNIC_ORIGIN_HISPANIC = get_fld(pRec,109,2);
m_ETHNIC_ORIGIN_NATIVE_AMERICAN = get_fld(pRec,111,2);
m_ETHNIC_ORIGIN_OTHER = get_fld(pRec,113,2);
stmp = get_fld(pRec,115,2);
if(stmp[0] == '1') m_MARITAL_STATUS_SINGLE = ("1"); else
m_MARITAL_STATUS_SINGLE = ("0");
if(stmp[0] == '2') m_MARITAL_STATUS_MARRIED = ("1"); else
m_MARITAL_STATUS_MARRIED = ("0");
if(stmp[0] == '3') m_MARITAL_STATUS_DIVORCED = ("1"); else
m_MARITAL_STATUS_DIVORCED = ("0");
if(stmp[0] == '4') m_MARITAL_STATUS_WIDOWED = ("1"); else
m_MARITAL_STATUS_WIDOWED = ("0");
if(stmp[0] == '5') m_MARITAL_STATUS_LWP = ("1"); else
m_MARITAL_STATUS_LWP = ("0");
if(stmp[0] == '6') m_MARITAL_STATUS_OTHER = ("1"); else
m_MARITAL_STATUS_OTHER = ("0");
m_ACOG_SYMPTOMS = get_fld(pRec,117,2);
stnp = get_fld(pRec,119,2);
if(stmp[0] == '0') m_VAGINAL_BLEEDING = ("0"); else
m_VAGINAL_BLEEDING = ("1");
if(stmp[0] == '1') m_VAGINAL_BLEEDING_TRACE = ("1"); else
m_VAGINAL_BLEEDING_TRACE = ("0");
if(stmp[0] == '2') m_VAGINAL_BLEEDING_MEDIUM = ("1"); else
m_VAGINAL_BLEEDING_MEDIUM = ("0");
if(stmp[0] == '3') m_VAGINAL_BLEEDING_GROSS = ("1"); else
m_VAGINAL_BLEEDING_GROSS = ("0");
if(stmp[0] == 0) m_VAGINAL_BLEEDING = ("0");
m_PATIENT_COMPLAINT_1 = get_fld(pRec,121,2);
m_PATIENT_COMPLAINT_2 = get_fld(pRec,123,2);
m_PATIENT_COMPLAINT_3 = get_fld(pRec,125,2);
m_PATIENT_COMPLAINT_4 = get_fld(pRec,127,2);
m_PATIENT_COMPLAINT_5 = get_fld(pRec,129,2);
m_PATIENT_COMPLAINT_6 = get_fld(pRec,131,2);
stmp_get_fld(pRec,133,2);
if(stmp[0] == '1') m_PATIENT_COMPLAINT_1_LT1 = ("1"); else
m_PATIENT_COMPLAINT_1_LT1 = ("0");
if(stmp[0] == '2') m_PATIENT_COMPLAINT_1_1_3 = ("1"); else
m_PATIENT_COMPLAINT_1_1_3 = ("0");
if(stmp[0] == '3') m_PATIENT_COMPLAINT_1_4_6 = ("1"); else
m_PATIENT_COMPLAINT_1_4_6 = ("0");
if(stmp[0] == '4') m_PATIENT_COMPLAINT_1_7_9 = ("1"); else
m_PATIENT_COMPLAINT_1_7_9 = ("0");
if(stmp[0] == '5') m_PATIENT_COMPLAINT_1_10_12 = ("1"); else
m_PATIENT_COMPLAINT_1_10_12 = ("0");
if(stmp[0] == '6') m_PATIENT_COMPLAINT_1_GT12 = ("1"); else
m_PATIENT_COMPLAINT_1_GT12 = ("0");
m_EGA_BY_SONO = get_fld(pRec,135,8);
m_EGA_BY_LMP = get_fld(pRec,143,8);
m_EGA_AT_SAMPLING = get_fld(pRec,151,8);
m_GRAVITY = get_fld(pRec,159,2);

```

```

m_PARITY = get_fld(pRec,161,2);
m_ABORTIONS = get_fld(pRec,163,2);
stmp = get_fld(pRec,165,2);
if(stmp[0] == '1') m_2_COMP_1 = ("1"); else m_2_COMP_1 = ("0");
if(stmp[0] == '2') m_2_COMP_2 = ("1"); else m_2_COMP_2 = ("0");
if(stmp[0] == '3') m_2_COMP_3 = ("1"); else m_2_COMP_3 = ("0");
m_0_COMP = get_fld(pRec,167,2);
m_1_COMP = get_fld(pRec,169,2);
m_2_COMP = get_fld(pRec,171,2);
m_3_COMP = get_fld(pRec,173,2);
m_4_COMP = get_fld(pRec,175,2);
m_5_COMP = get_fld(pRec,177,2);
m_6_COMP = get_fld(pRec,179,2);
stmp = get_fld(pRec,181,2);
if (stmp[0] == [0] , _MULTIPLE_GESTATION ("0"); else
m_MULTIPLE_GESTATION = ("1");
if(stmp[0] == '1') ("1"); m_MULTIPLE_GESTATION_TWINS = ("1"); else
m_MULTIPLE_GESTATION_TWINS = ("0");
if(stmp[0] == '2') m_MULTIPLE_GESTATION_TRIPLETS = ("1"); else
m_MULTIPLE_GESTATION_TRIPLETS = ("0");
if(stmp[0] == '3') m_MULTIPLE_GESTATION_QUADS = ("1"); else
m_MULTIPLE_GESTATION_QUADS = ("0");
if(stmp[0] == 0) m_MULTIPLE_GESTATION = ("0");
m_UTCERV_ABNORMALITY = get_fld(pRec,183,2);
m_CERVICAL_CERCLAGE = get_fld(pRec,185,2);
m_GESTATIONAL_DIABETES = get_fld(pRec,187,2);
m_HYPERTENSIVE_DISORDERS = get_fld(pRec,189,2);
stmp = get_fld(pRec,191,2);
if(stmp[0] == '0') m_DILITATION_UNKNOWN = ("1"); else
m_DILITATION_UNKNOWN = ("0");
if(stmp[0] == '1') m_DILITATION_LT1 = ("1"); else m_DILITATION LT1 =
("0");
if(stmp[0] == '2') m_DILITATION_1 = ("1"); else m_DILITATION_1 =
("0");
if(stmp[0] == '3') m_DILITATION_1_2 = ("1"); else m_DILITATION_1_2 =
("0");
if(stmp[0] == '4') m_DILITATION_2 = ("1"); else m_DILITATION_2 =
("0");
if(stmp[0] == '5') m_DILITATION_2_3 = ("1"); else m_DILITATION_2_3 =
("0");
if(stmp[0] == '6') m_DILITATION_3 = ("1"); else m_DILITATION_3 =
("0");
if(stmp[0] == '7') m_DILITATION_GT3 = ("1"); else m_DILITATION_GT3 =
("0");
stmp = get_fld(pRec,193,2);
if (stmp [0] == '1') m_CERVICAL_CONSISTANCY_FIRM = ("1") ; else
m_CERVICAL_CONSISTANCY_FIRM = ("0");
if (stmp[0] == '2') m_CERVICAL_CONSISTANCY_MOD = ("1"); else
m_CERVICAL_CONSISTANCY_MOD = ("0");
if (stmp[0] == '3') m_CERVICAL_CONSISTANCY_SOFT = ("1") else
m_CERVICAL_CONSISTANCY_SOFT = ("0");
m_ANTIBIOTICS = get_fld(pRec,195,2);
m_CORTICOSTEROIDS = get_fld(pRec,197,2);
m_TOYOLYTICS = get_fld(pRec,199,2);
m_INSULIN = get_fld(pRec,201,2);
m_ANTIHYPERTENSIVES = get_fld(pRec,203,2);
m_MEDICATIONS_NONE = get_fld(pRec,205,2);
m_MEDICATIONS_UNKNOWN = get_fld(pRec,207,2);
m_FFN_RESULT = get_fld(pRec,209,2);
m_NetPos1 = (double)atof(get_fld(pRec, 211, 20));
m_NetNeg1 = (double)atof(get_fld(pRec, 231, 20));

```

```

m_NetPos2 = (double)atof(get_fld(pRec, 251, 20));
m_NetNeg2 = (double)atof(get_fld(pRec, 271, 20));
m_NetPos3 = (double)atof(get_fld(pRec, 291, 20));
m_NetNeg3 = (double)atof(get_fld(pRec, 311, 20));

UpdateAllViews(NULL);
}

char* CPTDinpDoc::get_fld(char* pRec, int ofs, int len)
{
    int i;

    for( i = 0; i < len; i++) {
        fld[i] = pRec[ofs-1+i];
    }
    fld[len] = 0;
    for( i = len-1; i >= 0; i--) {
        if(fld[i] == ' ') {
            fld[i] = 0;
        } else {
            break;
        }
    }
    return fld;
}

CTime& CPTDinpDoc::get_time_fld(char* pRec, int iofs, int len)
{
    int i;
    int m,d,y;
    int ofs;

    for( i = 0; i < len; i++) {
        fld[i] = pRec(iofs-1+i);
    }
    for( i = len-1; i > 0; i--) {
        if(fld[i] == ' ')
            fld[i] = 0;
        } else {
            break;
        }
    }

    strcpy(tstr,fld);
    m = d = y = 0;
    ofs = 0;
    while(tstr[ofs] == ' ') ofs++; // skip spaces;
    m = atoi(&tstr[ofs]);
    while(tstr[ofs] >= '0' && tstr[ofs] <= '9') ofs++; // skip number
    while(tstr[ofs] == '/' || tstr[ofs] == '-') ofs++; // skip delimiter
    d = atoi(&tstr[ofs]);
    if (d == 0) d = 1;
    while(tstr[ofs] >= '0' && tstr[ofs] <= '9') ofs++; // skip number
    while(tstr[ofs] == '/' || tstr[ofs] == '-') ofs++; // skip delimiter
    y = atoi(&tstr[ofs]);
    if(y<100) y += 1900;

    CTime t(y,m,d,0,0,0);
    tim = t;
    return(tim);
}

```

}

```

void CPTDinpDoc::put_rec(char* pRec)
{
FILE *fp;
CString stmp;

    put_fld(pRec, m_LAB_ID ,1,12);
    put_fld(pRec, m_NAME_L ,13,24);
    put_fld(pRec, m_NAME_F ,37,24);
    put_fld(pRec, m_NAME_MI ,61,2);
    put_fld(pRec, m_DATE_OF_DATA_ENTRY ,63,10);           //time
    put_dbl_fld(pRec, m_PATIENT_AGE ,73,20);
    put_fld(pRec, m_DATE_OF_BIRTH ,93,10);

    //Stmp = " ";
    //if( m_ETHNIC_ORIGIN_WHITE == "1" ) stmp = "1";
    //if( m_ETHNIC_ORIGIN_BLACK == "1" ) stmp = "2";
    //if( m_ETHNIC_ORIGIN_ASIAN == "1" ) stmp = "3";
    //if( m_ETHNIC_ORIGIN_HISPANIC == "1" ) stmp = "4";
    //if( m_ETHNIC_ORIGIN_NATIVE_AMERICAN == ) stmp = "5";
    //if( m_ETHNIC_ORIGIN_OTHER == "1" ) stmp = "6";
    //put_fld(pRec, stmp,103,2);
    put_fld(pRec, m_ETHNIC_ORIGIN_WHITE ,103,2);
    put_fld(pRec, m_ETHNIC_ORIGIN_BLACK ,105,2);
    put_fld(pRec, m_ETHNIC_ORIGIN_ASIAN ,107,2);
    put_fld(pRec, m_ETHNIC_ORIGIN_HISPANIC ,109,2);
    put_fld(pRec, m_ETHNIC_ORIGIN_NATIVE_AMERICAN ,111, 2);
    put_fld(pRec, m_ETHNIC_ORIGIN_OTHER ,113,2);

    stmp = " ";
    if( m_MARITAL_STATUS_SINGLE == "1" ) stmp + "1";
    if( m_MARITAL_STATUS_MARRIED == "1" ) stmp + "2";
    if( m_MARITAL_STATUS_DIVORCED == "1" ) stmp + "3";
    if( m_MARITAL_STATUS_WIDOWED == "1" ) stmp + "4";
    if ( m_MARITAL_STATUS_LWP == "1" ) stmp + "5";
    if( m_MARITAL_STATUS_OTHER == "1" ) stmp + "6";
    put_fld(pRec, stmp,115,2);

    put_fld(pRec, m_ACOG_SYMPTOMS ,117,2);

    stmp = " ";
    if( m_VAGINAL_BLEEDING == "0" ) stmp = "0";
    if( m_VAGINAL_BLEEDING_TRACE == "1" ) stmp = "1";
    if( m_VAGINAL_BLEEDING_MEDIUM == "1" ) stmp = "2";
    if( m_VAGINAL_BLEEDING_GROSS == "1" ) stmp = "3";
    put_fld(pRec, stmp,119,2);

    put_fld(pRec, m_PATIENT_COMPLAINT_1 ,121,2);
    put_fld(pRec, m_PATIENT_COMPLAINT_2 ,123,2);
    put_fld(pRec, m_PATIENT_COMPLAINT_3 ,125,2);
    put_fld(pRec, m_PATIENT_COMPLAINT_4 ,127,2);
    put_fld(pRec, m_PATIENT_COMPLAINT_5 ,129,2);
    put_fld(pRec, m_PATIENT_COMPLAINT_6 ,131,2);

    stmp = " ";
    if( m_PATIENT_COMPLAINT_1_LT1 == "1" ) stmp = "1";
    if( m_PATIENT_COMPLAINT_1_1_3 == "1" ) stmp = "2";
    if( m_PATIENT_COMPLAINT_1_4_6 == "1" ) stmp = "3";

```

```

if( m_PATIENT_COMPLAINT_1_7_9 == "1" ) stmp = "4";
if( m_PATIENT_COMPLAINT_1_10_12 == "1" ) stmp = "5";
if( m_PATIENT_COMPLAINT_1_GT12 == "1" ) stmp = "6";
put_fld(pRec, stmp,133,2);

put_fld(pRec, m_EGA_BY_SONO ,135,8);
put_fld(pRec, m_EGA_BY_LMP ,143,8);
put_fld(pRec, m_EGA_AT_SAMPLING ,151,8);
put_fld(pRec, m_GRAVITY ,159,2);
put_fld(pRec, m_PARITY, 161,2);
put_fld(pRec, m_ABORTIONS, 163,2);

stmp = " ";
if( m_2_COMP_1 == "1" ) stmp = "1";
if( m_1_COMP_2 == "1" ) stmp = "2";
if( m_2_COMP_3 == "1" ) stmp = "3";
put_fld(pRec, stmp,165,2);

put_fld(pRec, m_0_COMP ,167,2);
put_fld(pRec, m_1_COMP ,169,2);
put_fld(pRec, m_2_COMP ,171,2);
put_fld(pRec, m_3_COMP ,173,2);
put_fld(pRec, m_4_COMP ,175,2);
put_fld(pRec, m_5_COMP ,177,2);
put_fld(pRec, m_6_COMP ,179,2);

stmp = " ";
if( m_MULTIPLE_GESTATION == "0" ) stmp = "0";
if( m_MULTIPLE_GESTATION_TWINS == "1" ) stmp = "1";
if( m_MULTIPLE_GESTATION_TRIPLETS == "1" ) stmp = "2";
if( m_MULTIPLE_GESTATION_QUADS == "1" ) stmp = "3";
put_fld(pRec, stmp,181,2);

put_fld(pRec, m_UTCERV_ABNORMALITY ,183,2);
put_fld(pRec, m_CERVICAL_CERCLAGE ,185,2);
put_fld(pRec, m_GESTATIONAL_DIABETES ,187,2);
put_fld(pRec, m_HYPERTENSIVE_DISORDERS ,189,2);

stmp = " ";
if( m_DILITATION_UNKNOWN == "1" ) stmp = "0";
if( m_DILITATION_LT1 == "1" ) stmp = "1";
if( m_DILITATION_1 == "1" ) stmp = "2";
if( m_DILITATION_1_2 == "1" ) stmp = "3";
if( m_DILITATION_2 == "1" ) stmp = "4";
if( m_DILITATION_2_3 == "1" ) stmp = "5";
if( m_DILITATION_3 == "1" ) stmp = "6";
if( m_DILITATION_GT3 == "1" ) stmp = "7";
put_fld(pRec, stmp,191,2);

stmp = " ";
if( m_CERVICAL_CONSISTANCY_FIRM == "1" ) stmp = "1";
if( m_CERVICAL_CONSISTANCY_MOD == "1" ) stmp = "2";
if( m_CERVICAL_CONSISTANCY_SOFT == "1" ) stmp = "3";
put_fld(pRec, stmp,193,2);

put_fld(pRec, m_ANTIBIOTICS ,195,2);
put_fld(pRec, m_CORTICOSTEROIDS ,197,2);
put_fld(pRec, m_TOXOLYTICS ,199,2);
put_fld(pRec, m_INSULIN ,201,2);
put_fld(pRec, m_ANTIHYPERTENSIVES ,203,2);
put_fld(pRec, m_MEDICATIONS_NONE ,205,2);

```

```

put_fld(pRec, m_MEDICATIONS_UNKNOWN ,207,2);
put_fld(pRec, m_FFN_RESULT ,209,2);
put_net_fld(pRec, m_NetPos1,211, 20);
put_net_fld(pRec, m_NetNeg1,231, 20);
put_net_fld(pRec, m_NetPos2,251, 20);
put_net_fld(pRec, m_NetNeg2,271, 20);
put_net_fld(pRec, m_NetPos3,291, 20);
put_net_fld(pRec, m_NetNeg3,311, 20);

fp = fopen(PathName,"r+b");
if(fp==NULL) {
    fp = fopen(PathName,"wb");
    if(fp!=NULL) {
        fwrite(Rec,sizeof (char), (REC_LENGTH+2L),fp);
        fflush(fp);
        fclose(fp);
    }
} else {
    f seek (fp, (long) ((REC_LENGTH+2L)*CurRecord),SEEK_SET);
    fwrite (pRec, sizeof (char), (REC_LENGTH+2L), fp);
    fflush(fp);
    fclose(fp);
}

UpdateAllViews(NULL);
}

void CPTDinpDoc::put_fld(char* pRec, CString& dat, int ofs, int len)
{
    int i;
    int fill;

    strcpy(fld,dat);
    fill = 0;

    for( i = 0; i < len; i++) {
        if (fld[i] == 0) fill = 1;
        if(fill==0) {
            pRec(ofs-1+i) = fld[i];
        } else {
            pRec(ofs-1+i) = (char)' ';
        }
    }
}

void CPTDinpDoc::put_dbl_fld (char* pRec, double dat, int ofs, int len)
{
    int i;

    sprintf(fld,"%20.4lf",dat);
    for( i = 0; i < len; i++) {
        pRec(ofs-1+i) = fld[i];
    }
}

void CPTDinpDoc::put_net_fld (char* pRec, double dat, int ofs, int len)
{
    int i;

```

```

        sprintf(fld,"%20.16lf",dat);
        for( i = 0; i < len; i++) {
            pRec[ofs-1+i] = fld[i];
        }
    }

void CPTDinpDoc: :put-time-fld (char* pRec, CTime& dat, int ofs, int len)

int i;
char *pfld;

    pfld = time2str(dat);
    strcat(pfld," ");
    for( i = 0; i < len; i++) {
        pRec[ofs-1+i] = pfld[i];
    }
}

void CPTDinpDoc::OnBldNetFile()
{
    FILE *fp;

    // Get the File Name
    CFileDialog Dlg (FALSE, "ndb", NULL, OFN_OVERWRITEPROMPT ,
        "NDB files (*.nbd)||*.ndb||");
    Dlg.m_ofn.lpstrTitle = "Open fixed length Network DataBase file";
    if( Dlg.DoModal() == IDOK ) {

        strcpy(NetName,Dlg.GetPathName());

        // open the new file
        fp = fopen(NetName,"wb");

        if(fp == NULL) {
            AfxMessageBox ("Could not open the neural network output
file!");
        } else {

            // build the record
            CurRecord = 0;
            HCURSOR hcurSave;
            hcurSave = SetCursor (LoadCursor (NULL, IDC_WAIT));

            while( CurRecord < NumRecords ) {

                // read the PTD record
                get_rec(Rec);

                // run the networks
                RunNets(CurRecord);

                // build the output record
                put_fld(NetRec, m_LAB_ID, 1, 12);
                put_net_fld(NetRec, m_NetPos1, 13, 20);
                put_net_fld(NetRec, m_NetNeg1, 33, 20);
                put_net_fld(NetRec, m_NetPos2, 53, 20);
                put_net_fld(NetRec, m_NetNeg2, 73, 20);
                put_net_fld(NetRec, m_NetPos3, 93, 20);
            }
        }
    }
}

```

```

        put_net_fld(NetRec, m_NetNeg3, 113, 20);
        NetRec[132] = (char)0x0d;
        NetRec[133] = (char)0x0a;

        // write the output record
        fwrite (NetRec, sizeof (char) 134, fp)

        // increment to the next PTD record
        CurRecord += 1;
    }

    close the new file
    fclose(fp);
    SetCursor(hcurSave);
}

}

}

void CPTDinpDoc::InitializeRec()
{
    // add one-time construction code here
    for(int i = 0; i < REC_LENGTH; i++) Rec[i] = (char)' ';
    Rec[REC_LENGTH] = (char)0x0d;
    Rec[REC_LENGTH + 1] = (char)0x0a;

    //CTime Dtime(1900,1,1,0,0,0);
    char* Dtime = "mm/dd/yy";

    m_LAB_ID = ("");
    m_NAME_L = ("");
    m_NAME_F = ("");
    m_NAME_MI = ("");
    m_DATE_OF_DATA_ENTRY = time2str(CTime::GetCurrentTime());
    m_PATIENT_AGE = 0.0;
    m_DATE_OF_BIRTH = Dtime;
    m_ETHNIC_ORIGIN_WHITE = ("");
    m_ETHNIC_ORIGIN_BLACK = ("");
    m_ETHNIC_ORIGIN_ASIAN = ("");
    m_ETHNIC_ORIGIN_HISPANIC = ("");
    m_ETHNIC_ORIGIN_NATIVE_AMERICAN = ("");
    m_ETHNIC_ORIGIN_OTHER = ("");
    m_MARITAL_STATUS_SINGLE = ("");
    m_MARITAL_STATUS_MARRIED = ("");
    m_MARITAL_STATUS_DIVORCED = ("");
    m_MARITAL_STATUS_WIDOWED = ("");
    m_MARITAL_STATUS_LWP = ("");
    m_MARITAL_STATUS_OTHER = ("");
    m_ACOG_SYMPTOMS = ("");
    m_PATIENT_COMPLAINT_1 = ("");
    m_PATIENT_COMPLAINT_1_1_3 = ("");
    m_PATIENT_COMPLAINT_1_10_12 = ("");
    m_PATIENT_COMPLAINT_1_4_6 = ("");
    m_PATIENT_COMPLAINT_1_7_9 = ("");
    m_PATIENT_COMPLAINT_1_GT12 = ("");
    m_PATIENT_COMPLAINT_1_LT1 = ("");
    m_VAGINAL_BLEEDING = ("");
    m_VAGINAL_BLEEDING_TRACE = ("");
    m_VAGINAL_BLEEDING_MEDIUM = ("");

```

```

m_VAGINAL_BLEEDING_GROSS = ("");
m_PATIENT_COMPLAINT_6 = ("");
m_PATIENT_COMPLAINT_3 = ("");
m_PATIENT_COMPLAINT_2 = ("");
m_PATIENT_COMPLAINT_5 = ("");
m_PATIENT_COMPLAINT_4 = ("");
m_EGA_BY_SONO = "ww.d";
m_EGA_BY_LMP = "ww.d";
m_EGA_AT_SAMPLING = "ww.d";
m_0_COMP = ("");
m_1_COMP = ("");
m_2_COMP = ("");
m_3_COMP = ("");
m_4_COMP = ("");
m_5_COMP = ("");
m_6_COMP = ("");
m_2_COMP_1 = ("");
m_2_COMP_2 = ("");
m_2_COMP_3 = ("");
m_GRAVITY = ("");
m_PARITY = ("");
m_ABORTIONS = ("");
m_MULTIPLE_GESTATION = ("");
m_MULTIPLE_GESTATION_TWINS = ("");
m_MULTIPLE_GESTATION_TRIPLETS = ("");
m_MULTIPLE_GESTATION_QUADS = ("");
m_UTCERV_ABNORMALITY = ("");
m_CERVICAL_CERCLAGE = ("");
m_GESTATIONAL_DIABETES = ("");
m_HYPERTENSIVE_DISORDERS = ("");
m_DILITATION_LT1 = ("");
m_DILITATION_1 = ("");
m_DILITATION_1_2 = ("");
m_DILITATION_2 = ("");
m_DILITATION_2_3 = ("");
m_DILITATION_3 = ("");
m_DILITATION_GT3 = ("");
m_DILITATION_UNKNOWN = ("");
m_CERVICAL_CONSISTANCY_FIRM = ("");
m_CERVICAL_CONSISTANCY_MOD = ("");
m_CERVICAL_CONSISTANCY_SOFT = ("");
m_ANTIIBIOTICS = ("");
m_CORTICOSTEROIDS = ("");
m_TOYOLYTICS = ("");
m_INSULIN = ("");
m_ANTIHYPERTENSIVES = ("");
m_MEDICATIONS_NONE = ("");
m_MEDICATIONS_UNKNOWN = ("");
m_FFN_RESULT = ("");
}

void CPTDinpDoc::LoadNets()
{
    // load eight networks for each consensus 1-8
    if(LoadNet(1,"ega6_0") != 1) {
        AfxMessageBox("Could not load ega6_0");
    }
    if(LoadNet(2,"ega6_1") != 2) {
        AfxMessageBox("Could not load ega6_1");
    }
}

```

```

}
if (LoadNet(3,"ega6_2") != 3) {
    AfxMessageBox("Could not load ega6_2");
}
if (LoadNet(4,"ega6_3") != 4) {
    AfxMessageBox("Could not load ega6_3");
}
if (LoadNet(5,"ega6_4") != 5) {
    AfxMessageBox("Could not load ega6_4");
}
if (LoadNet(6,"ega6_5") != 6) {
    AfxMessageBox("Could not load ega6_5");
}
if (LoadNet(7,"ega6_6") != 7) {
    AfxMessageBox("Could not load ega6_6");
}
if (LoadNet(8,"ega6_7") != 8) {
    AfxMessageBox("Could not load ega6_7");
}
}

// load eight networks for each consensus 9-16
if (LoadNet(9,"egad7f0") != 9) {
    AfxMessageBox("Could not load egad7f0");
}
if (LoadNet(10,"egad7f1") != 10) {
    AfxMessageBox("Could not load egad7f1");
}
if (LoadNet(11,"egad7f2") != 11) {
    AfxMessageBox("Could not load egad7f2");
}
if (LoadNet(12,"egad7f3") != 12) {
    AfxMessageBox("Could not load egad7f3");
}
if (LoadNet(13,"egad7f4") != 13) {
    AfxMessageBox("Could not load egad7f4");
}
if (LoadNet(14,"egad7f5") != 14) {
    AfxMessageBox("Could not load egad7f5");
}
if (LoadNet(15,"egad7f6") != 15) {
    AfxMessageBox("Could not load egad7f6");
}
if (LoadNet(16,"egad7f7") != 16) {
    AfxMessageBox("Could not load egad7f7");
}
}

// load eight networks for each consensus 17-24
if (LoadNet(17,"egad14f0") != 17) {
    AfxMessageBox("Could not load egad14f");
}
if (LoadNet(18,"egad14f1") != 18) {
    AfxMessageBox("Could not load egad14f1");
}
if (LoadNet(19,"egad14f2") != 19) {
    AfxMessageBox("Could not load egad14f2");
}
if (LoadNet(20,"egad14f3") != 20) {
    AfxMessageBox("Could not load egad14f3");
}
}

```

```

    if(LoadNet(21,"egad14f4") != 21) {
        AfxMessageBox("Could not load egad14f4");
    }
    if(LoadNet(22,"egad14f5") != 22) {
        AfxMessageBox("Could not load egad14f5");
    }
    if(LoadNet(23,"egad14f6") != 23) {
        AfxMessageBox("Could not load egad14f6");
    }
    if(LoadNet(24,"egad14f7") != 24) {
        AfxMessageBox("Could not load egad14f7");
    }
}

void CPTDinpDoc::FreeNets()
{
    for(int i = 1; i <= 24; i++) FreeNet(i);
}

void CPTDinpDoc::RunNets(long n)
{
    double Val, Vall, frac;

    Run first ega6 nets
    m_NetPos1 = 0.0;
    m_NetNeg1 = 0.0;
    for(int i = 1; i <=8; i++) {
        // build inputs from record
        Val = ((m_ETHNIC_ORIGIN_WHITE == "1")?1.0:0.0);
        PutInput(i,1,&Val);
        Val = ((m_MARITAL_STATUS_LWP == "1")?1.0:0.0);
        PutInput(i,2,&Val);
        Val = (double)atof(m_EGA_BY_SONO);
        frac = Val - floor(Val);
        Val = floor(Val) + (frac / 0.7);
        PutInput(i,3,&Val);
        //Val = (double)atof(m_EGA_BY_BEST);
        Val = (double)atof(m_EGA_BY_LMP);
        frac = Val - floor(Val);
        Val = floor(Val) + (frac / 0.7);
        Vall = (double)atof(m_EGA_BY_SONO);
        frac = Vall - floor(Vall);
        Vall = floor(Vall) + (frac / 0.7);
        if(Vall <= 13.0) {
            Val = Vall;
        } else {
            if(fabs(Val - Vall) > 2.0) {
                } else {
                    Val = Vall;
                }
        }
        PutInput(i,4,&Val);
        Val = (double)atof(m_EGA_AT_SAMPLING);
        frac = Val - floor(Val);
        Val = floor(Val) + (frac / 0.7);
        PutInput(i,5,&Val);
        Val = 0.0; // CD INTERP
        if( m_DILITATION_LT1 == "1" ) Val = 0.0;
        if( m_DILITATION_1 == "1" ) Val = 1.0;
        if( m_DILITATION_1_2 == "1" ) Val = 1.5;
    }
}

```

```

if( m_DILITATION_2 == "1" ) Val = 2.0;
if( m_DILITATION_2_3 == "1" ) Val = 2.0;
if( m_DILITATION_3 == "1" ) Val = 3.0;
if( m_DILITATION_GT3 == "1" ) Val = 3.0;
PutInput(i,6,&Val);
Val = 0.0; // Parity-PreTerm
if( m_2_COMP_1 == "1" ) Val = 1.0;
if( m_2_COMP_2 == "1" ) Val = 2.0;
if( m_2_COMP_3 == "1" ) Val = 3.0;
PutInput(i,7,&Val);
Val = ((m_VAGINAL_BLEEDING == "1")?1.0:0.0);
PutInput(i,8,&Val);
Val = 1.823197; // CERVICAL CONSISTANCY
if( m_CERVICAL_CONSISTANCY_FIRM == "1" ) Val = 1.0;
if( m_CERVICAL_CONSISTANCY_MOD == "1" ) Val = 2.0;
if( m_CERVICAL_CONSISTANCY_SOFT == "1" ) Val = 3.0;
PutInput(i,9,&Val);
Val = ((m_1_COMP == "1")?1.0:0.0);
PutInput(i,10,&Val);
Val = ((m_FFN_RESULT == "1")?1.0:0.0);
PutInput(i,11,&Val);

// iterate network
IterateNet(i);

// build consensus result
m_NetPos1 += GetState(i,3,1) / 8.0;
m_NetNeg1 += GetState(i,3,2) / 8.0;
}

m_NetVall = 25.0 * (m_NetPos1 - m_NetNeg1)

// Run first egad7f nets
m_NetPos2 = 0.0;
m_NetNeg2 = 0.0;
for (i = 9; i <= 16; i++)
    // build inputs from record
    Val = ((m_ETHNIC_ORIGIN_WHITE == "1")?1.0:0.0);
    PutInput(i,1,&Val);
    Val = ((m_PATIENT_COMPLAINT_1 == "1")?1.0:0.0);
    PutInput(i,2,&Val);
    Val = (double)atof(m_ABORTIONS);
    PutInput(i,3,&Val);
    Val = ((m_VAGINAL_BLEEDING == "1")?1.0:0.0);
    PutInput(i,4,&Val);
    Val = 0.0; //UC_INTERP
    if( m_PATIENT_COMPLAINT_1_LT1 == "1" ) Val = 1.0;
    if( m_PATIENT_COMPLAINT_1_1_3 == "1" ) Val = 2.0;
    if( m_PATIENT_COMPLAINT_1_4_6 == "1" ) Val = 3.0;
    if( m_PATIENT_COMPLAINT_1_7_9 == "1" ) Val = 4.0;
    if( m_PATIENT_COMPLAINT_1_10_12 == "1" ) Val = 5.0;
    if( m_PATIENT_COMPLAINT_1_GT12 == "1" ) Val = 6.0;
    PutInput(i,5,&Val);
    Val = ((m_0_COMP == "1")?1.0:0.0);
    PutInput(i,6,&Val);
    Val = ((m_FFN_RESULT == "1")?1.0:0.0);
    PutInput(i,7,&Val);

// iterate network
IterateNet(i);

```

```

        // build consensus result
        m_NetPos2 += GetState(i,3,1) / 8.0;
        m_NetNeg2 += GetState(i,3,2) / 8.0;
    }

    m_NetVal12 = 25.0 * (m_NetPos2-m_NetNeg2);

    // Run first egad14f nets
    m_NetPos3 = 0.0;
    m_NetNeg3 = 0.0;
    for(i = 17; i <=24; i++) {
        // build inputs from record
        Val = ((m_ETHNIC_ORIGIN_NATIVE-AMERICAN == "1")?1.0:0.0);
        PutInput(i,1,&Val);
        Val = ((m_MARITAL_STATUS_LWP == "1")?1.0:0.0);
        PutInput(i,2,&Val);
        Val = ((m_PATIENT_COMPLAINT_1 == "1")?1.0:0.0);
        PutInput(i,3,&Val);
        Val = 0.0; //CD_INTERP
        if( m_DILITATION_LT1 == "1" ) Val = 0.0;
        if( m_DILITATION_1 == "1" ) Val = 1.0;
        if( m_DILITATION_1_2 == "1" ) Val = 1.5;
        if( m_DILITATION_2 == "1" ) Val = 2.0;
        if( m_DILITATION_2_3 == "1" ) Val = 2.0;
        if( m_DILITATION_3 == "1" ) Val = 3.0;
        if( m_DILITATION_GT3 == "1" ) Val = 3.0;
        PutInput(i,4,&Val);
        Val = 0.0; //UC_INTERP
        if( m_PATIENT_COMPLAINT_1_LT1 == "1" ) Val = 1.0;
        if( m_PATIENT_COMPLAINT_1_1_3 == "1" ) Val = 2.0;
        if( m_PATIENT_COMPLAINT_1_4_6 == "1" ) Val = 3.0;
        if( m_PATIENT_COMPLAINT_1_7_9 == "1" ) Val = 4.0;
        if( m_PATIENT_COMPLAINT_1_10_12 == "1" ) Val = 5.0;
        if( m_PATIENT_COMPLAINT_1_GT12 == "1" ) Val = 6.0;
        PutInput(i,5,&Val);
        Val = ((m_0_COMP == "1")?1.0:0.0);
        PutInput(i,6,&Val);
        Val = ((m_FFN_RESULT == "1")?1.0:0.0);
        PutInput(i,7,&Val);

        // iterate network
        IterateNet(i);

        // build consensus result
        m_NetPos3 += GetState(i,3,1) / 8.0;
        m_NetNeg3 += GetState(i,3,2) / 8.0;
    }

    m_NetVal13 = 25.0 * (m_NetPos3-m_NetNeg3);
}

char* CPTDinpDoc::time2str( const CTime& tm )
{
    sprintf(tstr,"%d/%d/%d",tm.GetMonth(),tm.GetDay(),
(tm.GetYear()-1900));
    return tstr;
}

CTime& CPTDinpDoc::str2time( CString& str )
{
    int m,d,y;

```

```

int ofs;

strcpy(tstr,str);
m = d = y = 0;
ofs = 0;
while(tstr[ofs] == ' ') ofs++; // skip spaces;
m = atoi(&tstr[ofs]);
while(tstr[ofs] >= '0' && tstr[ofs] <= '9') ofs++; // skip number
while(tstr[ofs] == '/' || tstr[ofs] == '\\') ofs++; // skip delimiter
d = atoi(&tstr[ofs]);
while(tstr[ofs] >= '0' && tstr[ofs] <= '9') ofs++; // skip number
while(tstr[ofs] == '/' || tstr[ofs] == '\\') ofs++; // skip delimiter
y = atoi(&tstr[ofs]);
if(Y<100) y += 1900;

tim = CTime(y,m,d,0,0,0);
return(tim);
}

void CPTDinpDoc::OnRecGoto()
{
    CPTdGoto dlg;
    int i;

    // Define and run a dialog to select the search mode and rec number
etc.
    dlg.m_IDStr = IDStr;
    dlg.m_RecNum = CurRecord + 1;
    dlg.m_GotoMode = GotoMode;

    if(dlg.DoModal() == IDOK) {
        GotoMode = dlg.m_GotoMode;
        switch(GotoMode) {
            case 0:
                // record number
                CurRecord = dlg.m_RecNum - 1;
                if (CurRecord < 0) CurRecord = 0;
                if (CurRecord > NumRecords - 1 ) CurRecord = NumRecords -
1;
                get_rec(Rec);
                break;
            case 1:
                // ID string
                IDStr = dlg.m_IDStr;
                for (i = 0; i < NumRecords; i++) {
                    CurRecord = i;
                    get_rec(Rec);
                    if ( IDStr == m_LAB_ID ) break;
                }
                break;
            default:
                // Do nothing
                break;
        }
    }
}

void CPTDinpDoc::OnFileMruFile1()

```

```

{
    GetPrivateProfileString("Recent File List",          //lpszSection
                           "File1",                    //lpszEntry
                           "untitled",                  // lpszDefault
                           PathName,                    // lpszReturnBuffer
                           128,                          // cbReturnBuffer
                           "ptdinp.ini");               // lpszFilename

    get_file();
}
void CPTDinpDoc::OnFileMruFile2()
{
    GetPrivateProfileString ("Recent File List",        //lpszSection
                             "File2",                  //lpszEntry
                             "untitled",                // lpszDefault
                             PathName,                  // lpszReturnBuffer
                             128,                        // cbReturnBuffer
                             "ptdinp.ini");              // lpszFilename

    get_file();
}
void CPTDinpDoc::OnFileMruFile3()
{
    GetPrivateProfileString ("Recent File List",        //lpszSection
                             "File3",                  //lpszEntry
                             "untitled",                // lpszDefault
                             PathName,                  // lpszReturnBuffer
                             128,                        // cbReturnBuffer
                             "ptdinp.ini");              // lpszFilename

    get_file();
}
void CPTDinpDoc::OnFileMruFile4()
{
    GetPrivateProfileString ("Recent File List",        //lpszSection
                             "File4",                  //lpszEntry
                             "untitled",                // lpszDefault
                             PathName,                  // lpszReturnBuffer
                             128,                        // cbReturnBuffer
                             "ptdinp.ini");              // lpszFilename

    get_file();
}
void CPTDinpDoc::OnFileOpen()
{
    //FILE *fp;

    // Get the File Name
    CFileDialog Dlg (TRUE, "fdb", NULL, OFN_OVERWRITEPROMPT ,
                    "FDB iles (*.fdb) |*.fdb|");
    Dlg.m_ofn.lpstrTitle = "Open Fixed length DataBase file";
    if( Dlg.DoModal() == IDOK ) {
        strcpy(PathName, Dlg.GetPathName());
        AfxGetApp () ->AddToRecentFileList (PathName);
    }
}

```

```

        get_file();
#ifdef NOT
    CurRecord = 0;
    fp = fopen(PathName,"rb");

    if(fp==NULL) {
        fp = fopen(PathName,"wb");
        if(fp!=NULL) {
            fwrite(Rec, sizeof (char), (REC_LENGTH+2L), fp);
            fclose(fp);
        }
        NumRecords = 1;
        CurRecord = 0;
        InitializeRec();
        put_rec(Rec);
        get_rec(Rec);
    } else {
        CurRecord = 0;
        if (fread(Rec, sizeof(char), (REC_LENGTH+2 L), fp) == (REC_LENGTH+
2 L) {
            get_rec(Rec);
        }
        fseek(fp, 0L, SEEK_END);
        NumRecords = ftell(fp) / (REC_LENGTH+2L);
        fclose(fp);
    }
#endif
}

}

void CPTDinpDoc::get-file()
{
    FILE *fp;

    CurRecord = 0;
    fp = fopen(PathName,"rb")

    if(fp==NULL) {
        fp = fopen(PathName,"wb");
        if(fp!=NULL) {
            fwrite (Rec, sizeof (char), (REC_LENGTH+2L), fp);
            fclose(fp);
        }
        NumRecords = 1;
        CurRecord = 0;
        InitializeRec();
        put_rec(Rec);
        get_rec(Rec);
    } else {
        CurRecord = 0;
        if (fread(Rec, sizeof(char), (REC_LENGTH+2L), fp) == (REC_LENGTH+2L))
        {
            get_rec(Rec);
        }
        fseek(fp, 0L, SEEK_END);
        NumRecords = ftell(fp) / (REC_LENGTH+2L)
        fclose(fp);
    }
}

```

```
((CPTDinpApp*)AfxGetApp())->SaveMRU();
}
```

```
PTDinp.cpp : Defines the class behaviors for the application.
//
```

```
#include "stdafx.h"
#include "PTDinp.h"
```

```
#include "mainfrm.h"
#include "PTDidoc.h"
#include "PTDivw.h"
```

```
#ifdef _DEBUG
#undef THIS_FILE
static char BASED_CODE THIS_FILE[] = __FILE__;
#endif
```

```
////////////////////////////////////
////////////////////////////////////
// CPTDinpApp
```

```
BEGIN_MESSAGE_MAP(CPTDinpApp, CWinApp)
    //{AFX_MSG_MAP(CPTDinpApp)
    ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
    ON_COMMAND(ID_CLR_SUBFIELDS, OnClrSubfields)
    ON_COMMAND(ID_EDIT_MODE, OnEditMode)
    //}AFX_MSG_MAP
    // Standard file based document commands
    ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
    ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
    // Standard print setup command
    ON_COMMAND(ID_FILE_PRINT_SETUP, CWinApp::OnFilePrintSetup)
END_MESSAGE_MAP()
```

```
////////////////////////////////////
////////////////////////////////////
// CPTDinpApp construction
```

```
CPTDinpApp::CPTDinpApp()
{
    // TODO: add construction code here,
    // Place all significant initialization in InitInstance
    m_pDoc = NULL;
    EditMode = FALSE;
    ClearSubfields = FALSE;
}
```

```
////////////////////////////////////
////////////////////////////////////
// The one and only CPTDinpApp object
```

```
CPTDinpApp NEAR theApp;
```

```
////////////////////////////////////
////////////////////////////////////
// CPTDinpApp initialization
```

```

BOOL CPTDinpApp::InitInstance()
{
    // Standard initialization
    // If you are not using these features and wish to reduce the size
    // of your final executable, you should remove from the following
    // the specific initialization routines you do not need.

    SetDialogBkColor();      // Set dialog background color to gray
    LoadStdProfileSettings(); // Load standard INI file options
    (including MRU)

    // Register the application's document templates. Document templates
    // serve as the connection between documents, frame windows and
    views.

    CSingleDocTemplate* pDocTemplate;
    pDocTemplate = new CSingleDocTemplate(
        IDR_MAINFRAME,
        RUNTIME_CLASS(CPTDinpDoc),
        RUNTIME_CLASS(CMainFrame), // main SDI frame window
        RUNTIME_CLASS(CPTDinpView));
    AddDocTemplate(pDocTemplate);

    // create a new (empty) document
    OnFileNew();

    if (m_lpCmdLine[0] != '\0')
    {
        // TODO: add command line processing here
    }

    ClearSubfields = TRUE;
    // check the menu item
    CMenu* pMenu = Af xGetApp() ->m_pMainWnd->GetMenu()
    pMenu->CheckMenuItem(ID_CLR_SUB_FIELDS,MF_CHECKED)

    return TRUE;
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
    CAboutDlg();

    // Dialog Data
    //{{AFX_DATA(CAboutDlg)
    enum { IDD = IDD_ABOUTBOX }
    //}}AFX_DATA

    // Implementation
protected:
    virtual void DoDataExchange(CDataExchange* pDX); //DDX/DDV support
    //{{AFX_MSG(CAboutDlg)
    // No message handlers
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

```

```

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
   //{{AFX_DATA_INIT(CAboutDlg)
    //}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
   //{{AFX_DATA_MAP(CAboutDlg)
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
   //{{AFX_MSG_MAP(CAboutDlg)
    // No message handlers
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

// App command to run the dialog
void CPTDinpApp::OnAppAbout()
{
    CAboutDlg aboutDlg;

    aboutDlg.DoModal();
}
////////////////////////////////////
////////////////////////////////////
//CPTDinpApp commands

void CPTDinpApp::OnClrSubfields ( )
{
    if(ClearSubfields) = FALSE;
        // uncheck the menu item
        CMenu* pMenu = AfxGetApp ( )>m_pMainWnd->GetMenu ( );
        pMenu->CheckMenuItem (ID_CLR_SUBFIELDS, MF_UNCHECKED);

    } else {

        ClearSubfields = TRUE;
        // check the menu item
        CMenu* pMenu = AfxGetApp ( )->m_pMainWnd->GetMenu ( );
        pMenu->CheckMenuItem(ID_CLR_SUBFIELDS,MF_CHECKED);

    }

}

void CPTDinpApp::OnEditMode ( )
{
    if (Edit Mode) {

        EditMode = FALSE;
        // uncheck the menu item
        CMenu* pMenu = AfxGetAppo ( )>m_pMainWnd->GetMenu ( );
        pMenu->CheckMenuItem(ID_EDIT_MODE,MF_UNCHECKED);

    } else {

        EditMode = TRUE;
    }
}

```

```

        // check the menu item
        CMenu* pMenu = AfxGetApp ( )->m_pMainWnd->GetMenu ( );
        pMenu->CheckMenuItem(ID_EDIT_MODE,MF_CHECKED);
    }
}

void CPTDinpApp::SaveMRU ( )
{
    SaveStdProfileSettings ( );
}

; endoinp.def : Declares the module parameters for the application.

NAME                ENDOINP
DESCRIPTION 'IENDOINP Windows Application'
EXETYPE              WINDOWS

CODE                 PRELOAD MOVEABLE DISCARDABLE
DATA                 PRELOAD MOVEABLE MULTIPLE

HEAPSIZE              1024 ;          initial heap size
; Stack size is passed as argument to linker's /STACK option

// PTDinp.h : main header file for the PTDINP application
//

#ifndef _AFXWIN_H_
    #error include 'stdafx.h' before including this file for PCH
#endif
#include "resource.h" // main symbols
////////////////////////////////////
// CPTDinpApp:
// See PTDinp.cpp for the implementation of this class
//
#include "PTDidoc.h"

class CPTDinpApp : public CWinApp
{
public:
    CPTDinpApp( );

    CPTDinpDoc *m_pDoc;

    int NextDlgPage;
    int LastDlgPage;
    BOOL EditMode;
    BOOL ClearSubfields;

    CPTDinpDoc *GetDoco ( ) {
        return m_pDoc;
    }

    void SaveMRU( void );

// Overrides
    virtual BOOL InitInstance ( );

// Implementation

```

```

    //{AFX_MSG(CPTDinpApp)
    afx_msg void OnAppAbout ( );
    afx_msg void OnClrSubfields ( );
    afx_msg void OnEditMode ( );
    //ITAFX_MSG
    DECLARE_MESSAGE_MAP( )
};
/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////

// PTDivw.cpp : implementation of the CPTDinpView class

#include "stdafx.h"
#include "PTDinp.h"

#include "PTDidoc.h"
#include "PTDivw.h"

#include "PTDdlgl.h"

#ifdef _DEBUG
#undef THIS_FILE
static char BASED_CODE THIS_FILE[ ] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
// CPTDinpView

IMPLEMENT_DYNCREATE(CPTDinpView, CView)

BEGIN_MESSAGE_MAP(CPTDinpView, CView)
    //{AFX_MSG_MAP(CPTDinpView)
    ON_COMMAND(ID_DATA_EDIT, OnDataEdit)
    ON_COMMAND(ID_DATA_NEW, OnDataNew)
    // } }AFX_MSG_MAP
    // Standard printing commands
    ON_COMMAND(ID_FILE_PRINT, CView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_PREVIEW, CView::OnFilePrintPreview)
END_MESSAGE_MAP( )

/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
// CPTDinpView construction/destruction

CPTDinpView::CPTDinpView ( )
{
    // TODO: add construction code here
    ShowPrt = FALSE;
}

CPTDinpView::~CPTDinpView ( )
{
}

/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
// CPTDinpView drawing

void CheckOut(CDC* pDC, char *str, int xpos, int ypos, int val)
{
    pDC->TextOut( xpos, ypos, str, strlen(str)

```

```

        pDC->Rectangle (CRect ( xpos - 6*29, ypos - 2*29, xpos - 2*29,
ypos - 6*29));
        if(val) {
            CBrush brush( RGB(0,0,0) );
            pDC->FillRect( CRect ( xpos - 6*29, ypos - 2*29, xpos -
2*29, ypos 6*29), &brush)
;
//      pDC->MoveTo(xpos - 6*29, ypos - 2*29);
//      pDC->LineTo( xpos - 2*29, ypos - 6*29);
//      pDC->MoveTo(xpos - 6*29, ypos - 6*29);
//      pDC->LineTo( xpos - 2*29, ypos -2*29);

void CPTDinpView::OnDraw(CDC* pDC)
{
    CPTDinpDoc* pDoc = GetDocument ( );
    CPTDinpApp* pApp = ((CPTDinpApp*)AfxGetApp ( ));

    ASSERT_VALID(pDoc);
    CFont _font10, font12;
    TEXTMETRIC tm;
    int nHeight;
    int i;

    // TODO: add draw code for native data here
    pDC->SetMapMode(MM_TWIPS);
    font12.CreateFont(-240,0,0,0,500, FALSE, FALSE, 0, ANSI_CHARSET,
        OUT_DEFAULT_PRECIS, CLIP_DEFAULT_PRECIS,
        DEFAULT_QUALITY, DEFAULT_PITCH | FF_ROMAN, "Times New Roman");
    CFont* pOldFont = (CFont*) pDC->SelectObject(&font12);

    pDC->GetTextMetrics(&tm);
    nHeight = tm.tmHeight + tm.tmExternalLeading;

    char str[256];
    char name[64];

    //pDC->Rectangle (CRect (0, 0, 11505, -15105)); //    FULL PAGE RECT

    if(ShowPrt) {
        if(!pApp->EditMode) {
            sprintf(str, "                ADEZA DIAGNOSTIC SERVICES");
            pDC->TextOut( 2440, ((-1 * nHeight) - 720), str, strlen(str));

            sprintf (str, "Pre-Term Delivery Risk Assessment Software:");
            pDC->TextOut( 2440, ((-2 * nHeight) - 720), str, strlen(str) );

            sprintf(str, "                Test Report Form ");
            pDC->TextOut( 2440, ((-3 * nHeight) - 720), str, strlen(str));
        }
    } else {
        sprintf(str, "File: %s", pDoc->PathName());
        pDC->TextOut( 720, ((-1 * nHeight) - 720), str, strlen(str) );

        sprintf(str, "Current record: %ld", pDoc->CurRecord+1);
        pDC->TextOut( 720, ((-2 * nHeight) - 720), str, strlen(str) );

        sprintf(str, "Number of records: %ld", pDoc->NumRecords);
    }
}

```

```

    pDC->TextOut( 720, ((-3 * nHeight) - 720), str, strlen(str) );
}

if((ShowPrt && !pApp->EditMode) || (!ShowPrt)) {

    sprintf(str, " Lab ID #:");
    pDC->TextOut( 720, ((-5 * nHeight) - 720), str, strlen(str) );
    sprintf(str, "%s", PDoc->m_LAB_ID);
    pDC->TextOut( 4320, ((-5 * nHeight) - 720), str, strlen(str) );

    strcpy( name, PDoc->m_NAME_F);
    strcat( name, " ");
    strcat( name, PDoc->m_NAME_MI);
    strcat( name, " ");
    strcat( name, PDoc->m_NAME_L);
    sprintf(str, " Patient Name: ");
    pDC->TextOut( 720, ((-6 * nHeight) - 720), str, strlen(str) );
};

    sprintf(str, "%s", name);
    pDC->TextOut( 4320, ((-6 * nHeight) - 720), str,
strlen(str));

    pDoc->RunNets(pDoc->CurRecord);
    sprintf (str, " Pre-term Delivery Risk <34.6wks: ");
    pDC->TextOut( 720, H-7 * nHeight - 720), str, strlen(str) );
    sprintf(str, "%lf", pDoc->m_NetPos1);
    pDC->TextOut( 4320, ((-7 * nHeight) - 720), str, strlen(str) );

    sprintf(str, " Pre-term Delivery Risk <7 days: ");
    pDC->TextOut( 720, ((-8 * nHeight) - 720), str,
strlen(str) );
    sprintf(str, "%lf", pDoc->m_NetPos2);
    pDC->TextOut( 4320, ((-8 * nHeight) - 720), str,
strlen(str) );

    sprintf(str, " Pre-term Delivery Risk <14 days: ");
    pDC->TextOut( 720, ((-9 * nHeight) - 720), str,
strlen(str) );
    sprintf(str, "%lf", pDoc->m_NetPos3);
    pDC->TextOut( 4320, ((-9 * nHeight) - 720), str,
strlen(str) );

    //if(pDoc->m_ACOG_SYMPTOMS == "0") {
    //    sprintf (str, "DISCLAIMER APPLIES:");
    //    pDC->TextOut( 720, ((-12 * nHeight) - 720), str, strlen(str) );
    //};

    //}

    for( i = 5; i <= 10; i++) {
        pDC->MoveTo(700, ((-i * nHeight) - 720));
        pDC->LineTo(8640, ((-i * nHeight) - 720));
    }
    pDC->MoveTo(700, ((-5 * nHeight) - 720));
    pDC->LineTo(700, ((-10 * nHeight) - 720));
    pDC->MoveTo(4320, ((-5 * nHeight) - 720));
    pDC->LineTo(4320, ((10 * nHeight) - 720));
    pDC->MoveTo(8640, ((-5 * nHeight) - 720));
    pDC->LineTo(8640, ((10 * nHeight) - 720));
} else {

```

```

font10.CreateFont(-200,0,0,0,500, FALSE, FALSE, 0, ANSI_CHARSET,
    OUT_DEFAULT_PRECIS, CLIP_DEFAULT_PRECIS,
    DEFAULT_QUALITY, DEFAULT_PITCH I-FF_ROMAN, "Times New Roman");
pDC->SelectObject(&font10);
//pDC->Rectangle(CRect( 0,0,11505,-15105));
pDC->Rectangle(CRect( 1*29,-4*29,397*29,-22*29));
pDC->Rectangle(CRect( 1*29,-24*29,397*29,-42*29));
pDC->Rectangle(CRect( 1*29,-44*29,187*29,-95*29));
pDC->Rectangle(CRect( 187*29,-44*29,397*29,-95*29));
pDC->Rectangle(CRect( 1*29,-97*29,397*29,-114*29));
pDC->Rectangle(CRect( 1*29,-116*29,397*29,-218*29));
pDC->Rectangle(CRect( 1*29,-220*29,397*29,-240*29));
pDC->Rectangle(CRect( 1*29, -242*29, 187*29, -348*29) );
pDC->Rectangle(CRect( 187*29,-242*29,397*29,-348*29));
pDC->Rectangle(CRect( 1*29,-350*29,397*29,-375*29));
pDC->Rectangle(CRect( 1*29,-377*29,397*29,-404*29));
pDC->Rectangle(CRect( 1*29,-406*29,397*29,-425*29));
pDC->Rectangle(CRect( 1*29, -427*29, 397*29, -470*29) );
sprintf (str, "ADEZA Pre-Term Delivery Risk Assessment  ");
pDC->TextOut( 7*29f-10*29, str, strlen(str) );
sprintf(str,"Lab ID #: %s", pDoc->m - LAB_ID);
pDC->TextOut( 267*29,-10*29, str, strlen(str) );
sprintf(str,"PATIENT INFORMATION");
pDC->TextOut( 159*29,-29*29, str, strlen(str) );
strcpy( name, pDoc->m_NAME_L);
sprintf(str,"Name(las'E) %s", name);
pDC->TextOut( 7*29,-51*29, str, strlen(str) );
strcpy( name, pDoc->m_NAME_F);
sprintf(str,"First %s", name);
pDC->TextOut( 99*29,-51*29, str, strlen(str) );
strcpy( name, pDoc->m_NAME_MI);
sprintf(str,"M %s", name);
pDC->TextOut( 160*29,-51*29, str, strlen(str) );
sprintf(str,"DOB %s", pDoc->m_DATE_OF_BIRTH);

pDC->TextOut( 7*29,-69*29, str,      strlen(str) );
sprintf(str,"Ethnic origin:");
pDC->TextOut( 192*29,-48*29, str, strlen(str) );
Checkout(pDC, "Caucasian", 248*29,-48*29, (pDoc->m_ETHNIC_ORIGIN_WHITE ==
"1") );
Checkout (pDC, "African American", 298*29, -48*29,
(pDoc->m_ETHNIC_ORIGIN_BLACK ==
"1") );
Checkout(pDC, "Asian", 368*29,-48*29, (pDoc->m_ETHNIC_ORIGIN_ASIAN == "1")
Checkout(pDC, "Hispanic", 248*29,-59*29, (pDoc->m_ETHNIC_ORIGIN_HISPANIC =
= "1-")
);
Checkout (pDC, "Native American", 298*29,-59*29,
(pDoc->m_ETHNIC_ORIGIN_NATIVE_AME
RICAN == "1") );
Checkout(pDC, "Other", 368*29,-59*29, (pDoc->m_ETHNIC-ORIGIN-OTHER
sprintf(str,"Marital status:");
pDC->TextOut( 192*29,-72*29, str, strlen(str) );
Checkout(pDC, "Married", 248*29,-72*29, (pDoc->m_MARITAL_STATUS_MARRIED =
="1")
;
Checkout(pDC, "Single", 288*29,-72*29, (pDoc->m - MARITAL_STATUS_SINGLE
Checkout(pDC,"Divorced/Separated", 322*29,-72*f9,
(pDoc->m_MARITAL_STATUS_DIVORC
ED = "1") );

```

```

Checkout(pDC, "Widowed", 248*29, -83*29, (pDoc->m_MARITAL_STATUS_WIDOWED ==
"1") );
;
Checkout(PDC, "Living with partner", 293*29, -83*29,
(pDoc->m_MARITAL_STATUS_LWP=
= "1") );
Checkout(pDC, "Other", 368*29, -83*29, (pDoc->m_MARITAL_STATUS_OTHER ==
"1") );
sprintf (str, "PATIENT HISTORY AND CLINICAL INFORMATION");
pDC->TextOut( 117*29, -102*29, str, strlen(str) );
sprintf(str, "At the time of sampling was the patient experiencing signs and
symp
toms of possible preterm labor?");
pDC->TextOut( 7*29, -119*29, str, strlen(str) );
Checkout (pDC, "Yes", 339*29, -119*29, (pDoc->m_ACOG_SYMPTOMS == "1") );
Checkout(pDC, "No", 370*29, -119*29, (pDoc->m_ACOG_SYMPTOMS == "0") );
sprintf(str, "If yes, please mark all that apply. ");
pDC->TextOut( 7*29, -134*29, str, strlen(str) );
Checkout(pDC, "Uterine contractions with or without pain", 19*29, -145*29,
(pDoc->
m_PATIENT_COMPLAINT_1 == "1") );
sprintf(str, "Number/hr");
PDC->TextOut( 22*29, -158*29, str, strlen(str) );
Checkout (pDC, "<1", 73*29, -158*29, (pDoc->m_PATIENT_COMPLAINT_1_LT1 ==
"1") );
Checkout(pDC, "1-3", 105*29, -158*29, (pDoc->m_PATIENT_COMPLAINT_1_1_3 ==
"1") );
Checkout(pDC, "4-6", 137*29, -158*29, (pDoc->m_PATIENT_COMPLAINT_1_4_6 ==
"1") );
Checkout (pDC, "7-9", 73*29, -170*29, (pDoc->m_PATIENT_COMPLAINT_1_7_9 ==
"1") );
Checkout(pDC, "10-12", 105*29, -170*29, (pDoc->m_PATIENT_COMPLAINT_1_10_12 ==
"1") );

Checkout(pDC, ">12", 137*29, -170*29, (pDoc->m_PATIENT_COMPLAINT_1_GT12 ==
"1") );
Checkout(pDC, "Vaginal bleeding", 19*29, -181*29, (pDoc->m_VAGINAL_BLEEDING ==
"1") );

Checkout (pDC, "Trace" , 29*29, -194*29, (pDoc->m_VAGINAL_BLEEDING_TRACE ==
"1") );
Checkout(pDC, "Med", 64*29, -194*29, (pDoc->m_VAGINAL_BLEEDING_MEDIUM ==
"1") );

Checkout (pDC, "Gross", 94*29, -194*29, (pDoc->m_VAGINAL_BLEEDING_GROSS
== "1") );
Checkout(PDC, "Patient is not "feeling right"7111, 19*29, -205*29,
(pDoc->m_PATIENT_COMPLAINT_6 == "1") );
Checkout(pDC, "Bleeding during the second or third trimester",
167*29, -145*29, (pDoc->m_PATIENT_COMPLAINT_3 == "1") );
Checkout (pDC, "Intermittent lower abdominal pain, dull, low backpain,
pelvic pressure", 167*29, -157*29, (pDoc->m_PATIENT_COMPLAINT_2 == "1") );
Checkout (pDC, "Change in vaginal discharge amount, color, or consistency",
167
*29, -181*29, (pDoc->m_PATIENT_COMPLAINT_5 == "1") );

```

Chec kOut (pDC, 7Menstrual- like      crimping (with or without diarrhea)",  
 167\*29,-193\*29  
 9, (pDoc->m\_PATIENT\_COMPLAINT\_4 == "1") );  
 sprintf (str, "Gestational Age: EGA by first trimester sono %s ",  
 pDoc->m\_EGA\_BY\_S  
 ONO);  
 PDC->TextOut( 7\*29,-225\*29, str, strlen(str) );  
 sprintf (str, "EGA by LMP %s", pDoc->m\_EGA\_BY\_LMP);  
 pDC->TextOut( 197\*29,-225\*29, str, strlen(str) );  
 sprintf(str,"EGA at sampling %s",pDoc->m\_EGA\_AT\_SAMPLING);  
 pDC->TextOut( 287\*29,-225\*29, str, strlen(str) );  
 sprintf(str,"Previous Pregnancy: Please mark all that apply.");  
 pDC->TextOut( 7\*29,-249\*29, str, strlen(str) );  
 CheckOut(pDC, "Previous pregnancy, no complications", 19\*29,-260\*29,  
 (pDoc->m\_1\_COMP == "1") );  
 CheckOut(pDC, "History of Preterm delivery", 19\*29, -272\*29,  
 (pDoc->m\_2\_COMP == "1") );  
 sprintf(str,"if Yes, how many?");  
 PDC->TextOut( 22\*29,-284\*29, str, strlen(str) );  
 CheckOut(pDC,111", 97\*29,-284\*29, (pDoc->m\_2\_COMP\_1 == "1",)  
 CheckOut(pDC,"2", 122\*29,-284\*29, (pDoc->m\_2\_COMP\_2 == "1")  
 CheckOut(pDC,">211, 147\*29,-284\*29, (pDoc->m\_2\_COMP\_3 == "1") );  
 CheckOut(pDC,"History of Preterm PROM" 19\*29, - 269\*29, (pDoc->m\_3\_COMP ==  
 "1") );  
  
 CheckOut(pDC, "History of incompetent cervix", 19\*29,-308\*29,  
 (pDoc->m\_4\_COMP == "1") );  
 CheckOut(pDC, "History of PIH/preeclampsia", 19\*29,-320\*29, (pDoc->m\_5\_COMP  
 == "1" ) );  
 CheckOut(pDC, "History of SAB prior to 20 wks", 19\*29,-332\*29,  
 (pDoc->m\_6\_COMP == "1") );  
 CheckOut(pDC, "Multiple Gestation:", 209\*29,-272\*29,  
 (pDoc->m\_MULTIPLE\_GESTATION == "1"));  
 CheckOut(pDC, "Twins", 284\*29,-272\*29, (pDoc->m\_MULTIPLE\_GESTATION\_TWINS ==  
 "1") );  
 CheckOut(pDC, "Triplets", 317\*29,-272\*29, (pDoc ->m\_MULTI  
 PLE\_GESTATION\_TRIPLETS == "1") );  
 CheckOut(pDC, "Quads", 356\*29,-272\*29, (pDoc->m\_MULTIPLE\_GESTATION\_QUADS =  
 = "1") );  
 CheckOut(pDC, "Uterine or cervical abnormality", 209\*29,-284\*29,  
 pDoc->m\_UTCERV\_ABNORMALITY = "1" );  
 CheckOut(pDC,"Cerclage", 209\*29,-296\*29, (pDoc->m\_CERVICAL\_CERCLAGE == "1")  
 ); CheckOut(pDC, "Gestational Diabetes". 209\*29,-308\*-f9, (pDo  
 c7>m\_GESTATIONAL DIABETES == "1"));  
 CheckOut(pDC, "Hypertensive Disorders". 209\*29,-320\*29,  
 (pDoc->m\_HYPERTENSIVE\_DISORDERS == "1") );  
 sprintf (s tr, "Cervical Status immediately following sample  
 collection:");  
 pDC->TextOut( 7\*29,-352\*29, str, strlen(str) );  
 sprintf(str,"Dilatation (cm)");  
 PDC->TextOut( 9\*29,-364\*29, str, strlen(str) CheckOut(pDC,"<1",  
 64\*29,-364\*29, (pDoc->m\_DILITATION\_LT1 == "1")  
 CheckOut(pDC,'11", 85\*29,-364\*29, (pDoc->m\_DILITATION\_1 == "1") );  
 CheckOut(pDC,"1-2", 102\*29,-364\*29, (pDoc->m\_DILITATION\_1\_2 == "1") );  
 CheckOut(pDC, 1121', 123\*29, -364\*29, (pDoc->m\_DILATION\_2 == "1") );  
 CheckOut(pDC,"2-3", 140\*29,-364\*29f (pDoc-m\_DILITATION\_2\_3 == "1") );  
 CheckOut(pDC,'13", 163\*29,-364\*29, (pDoc->m\_DILATION\_3 == "1") );  
 CheckOut(pDC,">3'1, 180\*29,-364\*29, (pDoc->m\_DILATION\_GT3 == "1") );  
 CheckOut(pDC, "Unknown", 201\*29,-364\*29, (pDoc->m\_DILITATION\_UNKNOWN =  
 "1") );  
 sprintf(str,"Cervical consistancy");

```

pDC->TextOut( 249*29,-364*29, str, strlen(str) );
CheckOut(pDC, "Firm", 324*29,-364*29, (pDoc->m_CERVICAL_CONSISTANCY_FIRM =
= "1") );
CheckOut(pDC,"Mod", 350*29,-364*29, (pDoc->m_CERVICAL_CONSISTANCY_MOD = =
"1") );
Checkout (pDC, "Soft", 376*29,-364*29, (pDoc->m_CERVICAL_CONSISTANCY_SOFT =
= "1" ) );

sprintf (str, "Medications at Time of Test (check all that apply)");
pDC->TextOut( 7*29,-380*29, str, strlen(str) );
CheckOut(pDC, "Antibiotics", 23*29,-392*29, (pDoc->m_ANTIIBIOTICS = = "1")
);
CheckOut(pDC,"Corticosteroids", 76*29,-392*29, (pDoc->m_CORTICOSTEROIDS = =
"1") );

Checkout (pDC, "Tocolytis", 144*29, -392*29, (pDoc->m_TOYOLYTICS = = "1" );
CheckOut(pDC, "Insulin", 193*29,-392*29, (pDoc->m_INSULIN = = "1") );
Chec kOut (pDC, "Antihypertensives ", 234*29,-392*29,
(pDoc->m_ANTIHYPERTENSIVES = = "1") );
Checkout (pDC, "None", 311*29,-392*29, (pDoc->m_MEDICATIONS_NONE == "1") );
Checkout (pDC, "Unknown", 348*29,-392*29, (pDoc->m_MEDICATIZ5NS-UNKNOWN
sprintf (str, "Current Pregnancy: G: %s", pDoc->m_GRAVITY);
pDC->TextOut( 195*29,-249*29, str, strlen(str) );
sprintf(stz,"P: %s", pDoc->m_PARITY);
pDC->Textout( 303*29f-249*29, str, strlen(str) );
sprintf (str,"A: %s", pDoc->m_ABORTIONS);
pDC->Textout( 343*29,-249*29, str, strlen(str) );
sprintf (str, "Qualitative fFN Elisa Test Results:");
PDC->TextOut( 7*29,-411*29, str, strlen(str) );
CheckOut(pDC, "Positive", 144*29,-411*29, (pDoc->m_FFN_RESULT = = "1") );
CheckOut(pDC, "Negative", 234*29,-411*29, (pDoc->m_FFN_RESULT = ="0") );
sprintf (str, "Pre-term Delivery Risk <34.6wks: ");
pDC->TextOut( 7*29,-432*29, str, strlen(str) );
sprintf(st.r," %If ",pDoc->m - NetPos1);
pDC->TextOut( 150*29,-432*29, str, strlen(str) );
sprintf(str,"Pre-term Delivery Risk <7 days: ");
pDC->TextOut( 7*29, -444*29, str, strlen(str) );
sprintf(str," %If ",pDoc->m - NetPos2);
PDC->TextOut( 150*29, -444*29, str, strlen(str) );
sprintf (str, "Pre-term Delivery Risk <14 days: ");
pDC->TextOut( 7*29, -456*29, str, strlen(str) );
sprintf(str," %If ",pDoc->m NetPos3);
pDC->TextOut( 150*29, -456*29, str, strlen(str) );

//if(pDoc->m ACOG_SYNPTOMS = = "0") {
//    sprintf (str, "DISCLAIMER APPLIES: ");
//    pDC->TextOut( 7*29, -480*29, str, strlen(str) );
//}

}
pDC->SelectObject(pOldFont);

}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CPTDinPView printing

BOOL CPTDinPView::OnPreparePrinting(CPrintInfo* pInfo)
{
    // default preparation
    return DoPreparePrinting(pInfo);
}

```

```

}
void CPTDinpView::OnBeginPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
    // TODO: add extra initialization before printing
    ShowPrt = TRUE;
}

void CPTDinpView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
    // TODO: add cleanup after printing
    ShowPrt = FALSE;
    GetDocument ( )->UpdateAllViews(NULL);
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//CPTDinpView diagnostics

#ifdef _DEBUG
void CPfDinpView::AssertValid ( ) const
{
    CView::AssertValid ( );
}
I void CPTDinpView::Dump(CDumpContext& dc) const
{
    CView::Dump(dc);
}
CPTDinpDoc* CPTDinpView: :GetDocument ( ) // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CPTDinpDoc)));
    return (CPTDinpDoc*)m_pDocument;
}
#endif // _DEBUG

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// CPTDinpView message handlers

void CPTDinpView::Edit ( )
{
    CPTDinp dlg;

    int val;

    ((CPTDinpApp*)AfxGetApp ( ) )->NextDlgPage = 1;

    m_pSet = GetDocument ( );

    // initialize all the variables in the record to allow smooth cancel
    ///dlg.m - DATE_OF_DATA_ENTRY = m_P_Set->m_DATE_OF_DATA_ENTRY;
    ///dlg.m_PATIENT_AGE = m_pSet->m_PATIENT_AGE;
    ///CString m..DATE_OF_BIRTH;
    dlg.m_DATE_OF_BIRTH = m_pSet->m_DATE_OF_BIRTH;
    ///CString m_NAME_F;
    dlg.m_NAME_F = m_pSet ->m_NAME_F;
    ///CString m_NAME_L;
    ///dlg.m_NAME_L = m_pSet ->m_NAME_L;
    ///CString m_NAME_MI;
    dlg.m_NAME_MI = m_pSet->m_NAME_MI;

```

```

//BOOL      m_1_COMP;
//dlg.m_1 COMP = (m_pSet->m_1_COMP == "1");
//BOOL      m_2_COMP;
//dlg.m_2 COMP = (m_pSet->m_2_COMP == "1");
//BOOL      m_3_COMP;
//dlg.m_3 COMP = (m_pSet->m_3_COMP == "1");
//BOOL      m_4_COMP;
//dlg.m_4 COMP = (m_pSet->m_4_COMP == "1");
//BOOL      m_5_COMP;
//dlg.m_5 COMP = (m_pSet->m_5_COMP == "1");
//BOOL      m_6_COMP;
//dlg.m_6 COMP = (m_pSet->m_6_COMP == "1");
//BOOL      m_ACOG_N;
//dlg.m_ACOG_N = (m_pSet->m_ACOG_SYNPOTOMS == "0");
//BOOL      m_ACOG_Y;
//dlg.m_ACOG_Y = (m_pSet->m_ACOG_SYNPOTOMS == "1");
//BOOL      m_ANTIOTICS ==
//dlg.m_ANTIOTICS = (m_pSet->m_ANTIOTICS == "1");
//BOOL      m_AntiHyper;
//dlg.m_AntiHyper = (m_pSet->m_ANTIOTICS == "1");
//BOOL      m_CervCerclage;
//dlg.m_CervCerclage = (m_pSet->m_CERVICAL_CERCLAGE == "1");
//BOOL      m_CervFirm;
//dlg.m_CervFirm = (m_pSet->m_CERVICAL_CONSISTANCY_FIRM == "1");
//BOOL      m_CervMod;
//dlg.m_CervMod = (m_pSet->m_CERVICAL_CONSISTANCY_MOD == "1");
//BOOL      m_CervSoft;
//dlg.m_CervSoft = (m_pSet->m_CERVICAL_CONSISTANCY_SOFT == "1");
//BOOL      m_Corticosteroids;
//dlg.m_Corticosteroids = (m_pSet->m_CORTICOSTERIODS == "1");
//BOOL      m_Dilatation_1_2;
//dlg.m_Dilatation_1_2 = (m_pSet->m_DILATION_1_2 == "1");
//BOOL      m_Dilatation2;
//dlg.m_Dilatation2 = (m_pSet->m_DILATION_2 == "1");
//BOOL      m_Dilatation2_3;
//dlg.m_Dilatation2_3 = (m_pSet->m_DILATION_2_3 == "1");
//BOOL      m_Dilatation3;
//dlg.m_Dilatation3 = (m_pSet->m_DILATION_3 == "1");
//BOOL      m_DilatationGt3;
//dlg.m_DilatationGt3 = (m_pSet->m_DILATION_GT3 == "1");
//BOOL      m_Dilatation1;
//dlg.m_Dilatation1 = (m_pSet->m_DILATION_1 == "1");
//BOOL      m_DilatationLt1;
//dlg.m_DilatationLt1 = (m_pSet->m_DILATION_LT1 == "1");
//BOOL      m_DilatationUkn;
//dlg.m_DilatationUkn = (m_pSet->m_DILATION_UNKNOWN == "1");
//CString m_EGAatSample;
dlg.m_EGAatSample = m_pSet->m_EGA-AT-SAMPLING;
//CString m_EGAbbyLMP;
dlg.m_EGAbbyLMP = m_pSet->m_EGA_BY_LMP;
//CString m_EGAbbySONO;
dlg.m_EGAbbySONO = m_pSet->m_EGA_BY_SONO;
//BOOL      m_EthnicOriginAsian;
dlg.m_EthnicOriginAsian = m_pSet->m_ETHNIC_ORIGIN_ASIAN == "1";
//BOOL      m_EthnicOriginBlack;
dlg.m_EthnicOriginBlack = m_pSet->m_ETHNIC_ORIGIN_BLACK == "1";
//BOOL      m_EthnicOriginHispanic;
dlg.m_EthnicOriginHispanic = m_pSet->m_ETHNIC_ORIGIN_HISPANIC == "1";
//BOOL      m_EthnicNativeAmerican;
dlg.m_EthnicOriginNativeAmerican = m_pSet->m_ETHNIC_ORIGIN_NATIVEAMERICAN
== "1";

```

```

//BOOL      m_EthnicNativeOther;
dlg.m,_EthnicOriginNativeOther = m,_pSet->m_ETHNIC_ORIGIN_OTHER= =; "1");
//BOOL      m_EthnicNativeWhite;
dlg.m,_EthnicOriginNativeWhite = m,_pSet->m_ETHNIC_ORIGIN_WHITE= =; "1");
//BOOL      m_FFN_Neg;
dlg.m,_FFN_Neg = m,_pSet->m_FFN_RESULT= =; "0");
//BOOL      m_FFN_Pos;
dlg.m,_FFN_Pos = m,_pSet->m_FFN_RESULT= =; "1");
//BOOL      m_GestationDiabetes;
dlg.m,_GestationDiabetes = m,_pSet->m_GESTATIONAL_DIABETES = =; "1");
//BOOL      m_HypertensiveDisorders;
dlg.m,_HypertensiveDisorders = m,_pSet->m_HYPERTENSIVE_DISORDERS = =; "1");
//BOOL      m_Insulin;
dlg.m,_Insulin = m,_pSet->m_INSULIN = =; "1");
//CString m_LadID;
dlg.m,_LadID = m,_pSet->m_LAB_ID = =; "1");
//BOOL      m_MedicationNone;
dlg.m,_MedicationNone = m,_pSet->m_MEDICATIONS_NONE = =; "1");
//BOOL      m_MedicationUnknown;
dlg.m,_MedicationUnknown = m,_pSet->m_MEDICATIONS_UNKNOWN = =; "1");
//BOOL      m_MultipleGestationQuads;
dlg.m,_MultipleGestationQuads = m,_pSet->m_MULTIPLE_GESTATION_QUADS = =;
"1");
//BOOL      m_MultipleGestationTriplets;
dlg.m,_MultipleGestationTriplets = m,_pSet->m_MULTIPLE_GESTATION_TRIPLETS
= =; "1");
//BOOL      m_MultipleGestationTwins;
dlg.m,_MultipleGestationTwins = m,_pSet->m_MULTIPLE_GESTATION_TWINS = =;
"1");
//BOOL      m_MaritalStatusDivorced;
dlg.m,_MaritalStatusDivorced = m,_pSet->m_MARITAL_STATUS_DIVORCED = =;
"1");
//BOOL      m_MaritalStatusLWP;
dlg.m,_MaritalStatusLWP = m,_pSet->m_MARITAL_STATUS_LWP = =; "1");
//BOOL      m_MaritalStatusMarried;
dlg.m,_MaritalStatusMarried = m,_pSet->m_MARITAL_STATUS_MARRIED = =;
"1");
//BOOL      m_MaritalStatusOther;
dlg.m,_MaritalStatusOther = m,_pSet->m_MARITAL_STATUS_OTHER = =; "1");
//BOOL      m_MaritalStatusSingle;
dlg.m,_MaritalStatusSingle = m,_pSet->m_MARITAL_STATUS_SINGLE = =; "1");
//BOOL      m_MaritalStatusWidowed;
dlg.m,_MaritalStatusWidowed = m,_pSet->m_MARITAL_STATUS_WIDOWED = =;
"1");
//BOOL      m_MultipleGestation;
dlg.m,_MultipleGestation = m,_pSet->m_MULTIPLE_GESTATION= =; "1");
//BOOL      m_PatientComp1;
dlg.m,_PatientComp1 = m,_pSet->m_PATIENT_COMPLAINT_1= =; "1");
//BOOL      m_PatientComp2;
dlg.m,_PatientComp2 = m,_pSet->m_PATIENT_COMPLAINT_2= =; "1");
//BOOL      m_PatientComp3;
dlg.m,_PatientComp3 = m,_pSet->m_PATIENT_COMPLAINT_3= =; "1");
//BOOL      m_PatientComp4;
dlg.m,_PatientComp4 = m,_pSet->m_PATIENT_COMPLAINT_4= =; "1");
//BOOL      m_PatientComp5;
dlg.m,_PatientComp5 = m,_pSet->m_PATIENT_COMPLAINT_5= =; "1");
//BOOL      m_PatientComp6;
dlg.m,_PatientComp6 = m,_pSet->m_PATIENT_COMPLAINT_6= =; "1");
//BOOL      m_Tocolytics;

```

```

dlg.m_Tocolytics = (m_pSet->m_TOYOLYTICS
//BOOL      m_UtCervAbnormal,
dlg.m_UtCervAbnormal = (m_pSet->m_UTCERV_ABNORMALITY == "1");
//BOOL      m_VaginalBleeding;
dlg.m_VaginalBleeding = (m_pSet->m_VAGINAL_BLEEDING == "1");
//BOOL      m_VaginalBleedingGross;
dlg.m_VaginalBleedingGross = (m_pSet->m_VAGINAL_BLEEDING_GROSS
//BOOL      m_VaginalBleedingMed;
dlg.m_VaginalBleedingMed = (m_pSet->m_VAGINAL_BLEEDING_MEDIUM
//BOOL      m_VaginalBleedingTrace;
dlg.m_VaginalBleedingTrace = (m_pSet->m_VAGINAL_BLEEDING_TRACE
//BOOL      m_2_COMP_1;
dlg.m_2_COMP_1 = (m_pSet->m_2_COMP_1 == "1");
//BOOL      m_2_COMP_2;
//dlg.m_2_COMP_2 = (m_pSet->m_2_COMP_2 == "1");
//BOOL      m_2_COMP_3;
//dlg.m_2_COMP_3 = (m_pSet->m_2_COMP_3 == "1");
//CString m_ABORTIONS;
dlg.m_ABORTIONS = m_pSet->m_ABORTIONS;
//CString m_GRAVITY;
dlg.m_GRAVITY = m_pSet->m_GRAVITY;
//CString m_PARITY;
dlg.m_PARITY = m_pSet->m_PARITY;
//BOOL      m_PatComp1_1_3;
dlg.m_PatComp1_1_3 = (m_pSet->m_PATIENT_COMPLAINT_1_1_3 == "1");
//BOOL      m_PatComp1_10_12;
dlg.m_PatComp1_10_12 = (m_pSet->m_PATIENT_COMPLAINT_1_10_12 == "1");
//BOOL      m_PatComp1_4_6;
dlg.m_PatComp1_4_6 = (m_pSet->m_PATIENT_COMPLAINT_1_4_6 == "1");
//BOOL      m_PatComp1_7_9;
dlg.m_PatComp1_7_9 = (m_pSet->m_PATIENT_COMPLAINT_1_7_9 == "1");
//BOOL      m_PatComp1_GT12;
dlg.m_PatComp1_GT12 = (m_pSet->m_PATIENT_COMPLAINT_1_GT12 == "1");
//BOOL      m_PatComp1_LT1;
dlg.m_PatComp1_LT1 = (m_pSet->m_PATIENT_COMPLAINT_1_LT1 == "1");

if(dlg.DoModal() == IDOK) {

    //dlg.m_DATE_OF_DATA_ENTRY = m_pSet->m_DATE_OF_DATA_ENTRY;
    //dlg.m_PATIENT_AGE = m_pSet->m_PATIENT_AGE;
    //CString m_DATE_OF_BIRTH;
    m_pSet->m_DATE_OF_BIRTH = dlg.m_DATE_OF_BIRTH;
    //CString m_NAME_F;
    m_pSet->m_NAME_F = dlg.m_NAME_F;
    //CString m_NAME_L;
    m_pSet->m_NAME_L = dlg.m_NAME_L;
    //CString m_NAME_MI;
    m_pSet->m_NAME_MI = dlg.m_NAME_MI;
    //BOOL      m_1_COMP;
    m_pSet->m_1_COMP = (dlg.m_1_COMP?"1":"0");
    //BOOL      m_2_COMP;
    m_pSet->m_2_COMP = (dlg.m_2_COMP?"1":"0");
    //BOOL      m_3_COMP;
    m_pSet->m_3_COMP = (dlg.m_3_COMP?"1":"0");
    //BOOL      m_4_COMP;
    m_pSet->m_4_COMP = (dlg.m_4_COMP?"1":"0");
    //BOOL      m_5_COMP;
    m_pSet->m_5_COMP = (dlg.m_5_COMP?"1":"0");
    //BOOL      m_6_COMP;
    m_pSet->m_6_COMP = (dlg.m_6_COMP?"1":"0");
    //BOOL      m_ACOG_N;

```

```

m_pSet->m_ACOG_SYMPTOMS = (dlg.m_ACOG-N?"0":" ");
//BOOL      m_ACOG_Y;
m_pSet->m_ACOG_SYMPTOMS =
(dlg.m_ACOG_Y?"1":m_pSet->m_ACOG_SYMPTOMS);
//BOOL      m_Antibiotics;
m_pSet->m_ARTIBIOTICS = (dlg.m-Antibiotics?"1":"0");
//BOOL      m_AntiHyper;
m_pSet->m_ANTIHYPERTENSIVES = (dlg.m_AntiHyper?"1":"0");
//BOOL      m_CervCerclage;
m_pSet->m_CIRVICAL_CERCLAGE = (dlg.m_CervCerclage?"1":"0");
//BOOL      m_CervFirm;
m_pSet->m_CERVICAL_CONSISTANCY_FIRM = (dlg.m_CervFirm?"1":"0");
//BOOL      m_CervMod;
m_pSet->m_CERVICAL_CONSISTANCY_MOD = (dlg.m_CervMod?"1":"0");
//BOOL      m_CervSoft;
m_pSet->m_CERVICAL_CONSISTANCY_SOFT = (dlg.m_CervSoft?"1":"0");
//BOOL      m_Corticosteroids;
m_pSet->m_CORTICOSTEROIDS = (dlg.m_Corticosteroids?"1":"0");
//BOOL      m_Dililation1_2;
m_pSet->m_DILITATION_1_2 = (dlg.m_Dililation1_2?"1":"0");
//BOOL      m_Dililation2;
m_pSet->m_DILITATION_2 = (dlg.m_Dililation2?"1":"0");
//BOOL      m_Dililation2-3;
m_pSet->m_DILITATION_2_3 = (dlg.m_Dililation2_3?"1":"0");
//BOOL      m_Dililation3;
m_pSet->m_DILITATION_3 = (dlg.m_Dililation3?"1":"0");
//BOOL      m_DililationGt3;
m_pSet->m_DILITATION_GT3 = (dlg.m_DililationGt3?"1":"0");
//BOOL      m_Dililation1;
m_pSet->m_DILITATION_1 = (dlg.m_Dililation1?"1":"0");
//BOOL      m_DililationLt1;
m_pSet->m_DILITATION_LT1 = (dlg.m_DililationLt1?"1":"0");
//BOOL      m_DililationUkn;
m_pSet->m_DILITATION_UNKNOWN = (dlg.m_DililationUkn?"1":"0");
//CString m_EGAatSample;
m_pSet->m_EGA_AT_SAMPLING = dlg.m_EGAatSample;
//CString m_EGAbyLMP;
m_pSet->m_EGA_BY_LMP = dlg.m_EGAbyLMP;
//CString m_EGAbySONO;
m_pSet->m_EGA_BY_SONO = dlg.m_EGAbySONO;
//BOOL      m_EthnicOriginAsian;
m_pSet->m_EYHNIC_ORIGIN_ASIAN =
(dlg.m_EthnicOriginAsian?"1":"0");
//BOOL      m_EthnicOriginBlack;
m_pSet->m_ETHNIC_ORIGIN_BLACK =
(dlg.m_EthnicOriginBlack?"1":"0");
//BOOL      m_EthnicOriginHispanic;
m_pSet->m_ETHNIC_ORIGIN_HISPANIC =
(dlg.m_EthnicOriginHispanic?"1":"0");
//BOOL      m_EthnicOriginNativeAmerican;
m_pSet->m_ETHNIC_ORIGIN_NATIVE_AMERICAN =
(dlg.m_EthnicOriginNativeAmerican?"1":"0");
//BOOL      m_EthnicOriginOther;
m_pSet->m_ETHNIC_ORIGIN_OTHER =
(dlg.m_EthnicOriginOther?"1":"0");
//BOOL      m_EthnicOriginWhite;
m_pSet->m_ETHNIC_ORIGIN_WHITE =
(dlg.m_EthnicOriginWhite?"1":"0");
//BOOL      m_FFN_Neg;
m_pSet->mFFN_RESULT = (dlg.m_FFN_Neg?"0":" ");
//BOOL      m_FFN_Pos;

```

```

m_pSet->m_FFN_RESULT =
(dlg.m_FFN_Pos?"1":m_pSet->m_FFN_RESULT);
//BOOL      m_GestationalDiabetes;
m_pSet->m_GESTATIONAL_DIABETES =
(dlg.m_GestationalDiabetes?"1":"0");
//BOOL      m_HypertensiveDisorders;
m_pSet->m_HYPERTENSIVE_DISORDERS =
(dlg.m_HypertensiveDisorders?"1":"0");
//BOOL      m_Insulin;
m_pSet->m_INSULIN = (dlg.m_Insulin?"1":"0");
//CString m_LadID;
m_pSet->m_LAB_ID = dlg.m_LadID;
//BOOL      m_MedicationNone;
m_pSet->m_MEDICATIONS_NONE = (dlg.m_MedicationNone?"1":"0");
//BOOL      m_MedicationUnknown;
m_pSet->m_MEDICATIONS_UNKNOWN =
(dlg.m_MedicationUnknown?"1":"0");
//BOOL      m_MultipleGestationQuads;
m_pSet->m_MULTIPLE_GESTATION_QUADS =
(dlg.m_MultipleGestationQuads?"1":"0");
//BOOL      m_MultipleGestationTriplets;
m_pSet->m_MULTIPLE_GESTATION_TRIPLETS =
(dlg.m_MultipleGestationTriplets?"1":"0");
//BOOL      m_MultipleGestationTwins;
m_pSet->m_MULTIPLE_GESTATION_TWINS =
(dlg.m_MultipleGestationTwins?"1":"0");
//BOOL      m_MaritalStatusDivorced;
m_pSet->m_MARITAL_STATUS_DIVORCED =
(dlg.m_MaritalStatusDivorced?"1":"0");
//BOOL      m_MaritalStatusLWP;
m_pSet->m_MARITAL_STATUS_LWP =
(dlg.m_MaritalStatusLWP?"1":"0");
//BOOL      m_MaritalStatusMarried;
m_pSet->m_MARITAL_STATUS_MARRIED =
(dlg.m_MaritalStatusMarried?"1":"0");
//BOOL      m_MaritalStatusOther;
m_pSet->m_MARITAL_STATUS_OTHER = (dlg.m_MaritalStatusOther?"1":
"0");
//BOOL      m_MaritalStatusSingle;
m_pSet->m_MARITAL_STATUS_SINGLE =
(dlg.m_MaritalStatusSingle?"1":"0");
//BOOL      m_MaritalStatusWidowed;
m_pSet->m_MARITAL_STATUS_WIDOWED =
(dlg.m_MaritalStatusWidowed?"1":"0");
//BOOL      m_MultipleGestation;
m_pSet->m_MULTIPLE_GESTATION = (dlg
m_MultipleGestation?"1":"0");
//BOOL      m_PatientComp1;
m_pSet->m_PATIENT_COMPLAINT_1 = (dlg.m_PatientComp1?"1":"0");
//BOOL      m_PatientComp2;
m_pSet->m_PATIENT_COMPLAINT_2 = (dlg.m_PatientComp2?"1":"0");
//BOOL      m_PatientComp3;
m_pSet->m_PATIENT_COMPLAINT_3 = (dlg.m_PatientComp3?"1":"0");
//BOOL      m_PatientComp4;
m_pSet->m_PATIENT_COMPLAINT_4 = (dlg.m_PatientComp4?"1":"0");
//BOOL      m_PatientComp5;
m_pSet->m_PATIENT_COMPLAINT_5 = (dlg.m_PatientComp5?"1":"0");
//BOOL      m_PatientComp6;
m_pSet->m_PATIENT_COMPLAINT_6 = (dlg.m_PatientComp6?"1":"0");
//BOOL      m_Tocolytics;
m_pSet->m_TOCOLYTICS = (dlg.m_Tocolytics?"1":"0");

```

```

        //BOOL      m_UtCervAbnormal;
        m_pSet->m_UTCERV_ABNORMALITY = (dlg.m_UtCervAbnormal?"1":"0");
        //BOOL      m_VaginalBleeding;
        m_pSet->m_VAGINAL_BLEEDING = (dlg.m_VaginalBleeding?"1":"0");
        //BOOL      m_VaginalBleedingGross;
        m_pSet->m_VAGINAL_BLEEDING_GROSS =
(dlg.m_VaginalBleedingGross?"1":"0");
        //BOOL      m_VaginalBleedingMed;
        m_pSet->m_VAGINAL_BLEEDING_MEDIUM =
(dlg.m_VaginalBleedingMed?"1":"0");
        //BOOL      m_VaginalBleedingTrace;
        m_pSet->m_VAGINAL_BLEEDING_TRACE =
(dlg.m_VaginalBleedingTrace?"1":"0");
        //BOOL      m_2_COMP_1;
        m_pSet->m_2_COMP_1 = (dlg.m_2_COMP_1?"1":"0");
        //BOOL      m_2_COMP_2;
        m_pSet->m_2_COMP_2 = (dlg.m_2_COMP_2?"1":"0");
        //BOOL      m_2_COMP_3;
        m_pSet->m_2_COMP_3 = (dlg.m_2_COMP_3?"1":"0");
        //CString m_ABORTIONS;
        m_pSet->m_ABORTIONS = dlg.m_ABORTIONS;
        //CString m_GRAVITY;
        m_pSet->m_GRAVITY = dlg.m_GRAVITY;
        val = atoi(m_pSet->m_GRAVITY);
        if(val == 0) {
            m_pSet->m_0_COMP = "1";
        } else {
            m_pSet->m_0_COMP = "0";
        }
        //CString m_PARITY;
        m_pSet->m_PARITY = dlg.m_PARITY;
        //BOOL      m_PatComp1_1_3;
        m_pSet->m_PATIENT_COMPLAINT_1_1_3 =
(dlg.m_PatComp1_1_3?"1":"0");
        //BOOL      m_PatComp1_10_12;
        m_pSet->m_PATIENT_COMPLAINT_1_10_12 =
(dlg.m_PatComp1_10_12?"1":"0");
        //BOOL      m_PatComp1_4_6;
        m_pSet->m_PATIENT_COMPLAINT_1_4_6 =
(dlg.m_PatComp1_4_6?"1":"0");
        //BOOL      m_PatComp1_7_9;
        m_pSet->m_PATIENT_COMPLAINT_1_7_9 =
(dlg.m_PatComp1_7_9?"1":"0");
        //BOOL      m_PatComp1_GT12;
        m_pSet->m_PATIENT_COMPLAINT_1_GT12 =
(dlg.m_PatComp1_GT12?"1":"0");
        //BOOL      m_PatComp1_LT1;
        m_pSet->m_PATIENT_COMPLAINT_1_LT1 =
(dlg.m_PatComp1_LT1?"1":"0");

        // generate the net fields
        m_pSet->RunNets(m_pSet->CurRecord);

        // write the record to the file
        m_pSet->put_rec(m_pSet->Rec);
    }
}

int CPTDinView::str2int( CString& str )

```

```

{
    if(str == "0") return 2;
    if(str == "1") return 1;
    if(str == "2") return 0;
    return -1;
}

char* CPTDinpView::int2str( int val )
{
    if(val == 0) return "2";
    if(val == 1) return "1";
    if(val == 2) return "0";
    return " ";
}

int CPTDinpView::yn2int( CString& str )
{
    if(str == "0") return 1;
    if(str == "1") return 0;
    return -1;
}

char* CPTDinpView::int2yn( int val )
{
    if(val == 0) return "1";
    if(val == 1) return "0";
    return " ";
}

void CPTDinpView::OnDataEdit( )
{
    CPTDinpDoc*pDoc = GetDocument( );
    FILE *fp;

    fp = fopen(pDoc->PathName,"rb");
    if(fp!=NULL) {
        fclose(fp);
    } else {
        CFileDialog Dlg (TRUE, "fdb", NULL, OFN_OVERWRITEPROMPT ,
            "FDB files (*.fdb) ??*");
        Dlg.m_ofn.lpstrTitle = "Open Fixed length DataBase file";
        if( Dlg.DoModal() == IDOK ) {
            strcpy (pDoc->PathName, Dlg.GetPathName ());
            fp = fopen(pDoc->PathName,"rb");
            if(fp!=NULL) {
                AfxMessageBox("Unable to open Database File!");
                return;
            }
            pDoc->CurRecord = 0;
            fseek(fp,0L,SEEK_END);
            pDoc->NumRecords = ftell(fp) / (REC_LENGTH+2L);
            fclose(fp);
        }
    }

    Edit( );
}

void CPTDinpView::OnDataNew( )
{

```

```

FILE *fp;

CPTDinpDoc* pDoc = GetDocument( );

create a new record
fp = fopen(pDoc->PathName,"ab");
if(fp!=NULL) {
    fwrite (pDoc->Rec, sizeof (char), (REC_LENGTH + 2L), fp)
    fclose(fp);
}
pDoc->InitializeRec();
pDoc->NumRecords += 1;
pDoc->CurRecord = pDoc->NumRecords - 1;
pDoc->put_rec(pDoc->Rec);
// edit the new record
pDoc->get_rec(pDoc->Rec);
Edit( );
}

// PTDIVW.h : interface of the CPTDinpView class
//
/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////

class CPTDinpView : public CView

protected: // create from serialization only
    CPTDinpView( );
    DECLARE_DYNCRF.ATE(CPTDinpView)

//Attributes
public:
    CPTDinpDoc* GetDocument ( );

    BOOL ShowPrt;

    CPTDinpDoc* m_pSet;
    void Edit( void );

// Operations
public:
    // conversions for dialogs
    int str2int( CString& str );
    char* int2str( int val );
    int yn2int( CString& str );
    char* int2yn( int val );

// Implementation
public:
    virtual ~CPTDinpView( );
    virtual void OnDraw(CDC* PDC); // overridden to draw this view
#ifdef _DEBUG
    virtual void AssertValid( ) const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:

    // Printing support
    virtual BOOL OnPreparePrinting(CPrintInfo* pInfo);
    virtual void OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo);

```

```

        virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo);

// Generated message map functions
protected:
   //{{AFX_MSG(CPTDinpView)
    afx_msg void OnDataEdit ( );
    afx_msg void OnDataNew ( );
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP ( )
};

#ifndef _DEBUG // debug version in PTDivw.cpp
inline CPTDinpDoc* CPTDinpView:: GetDocument ( )
{ return (CPTDinpDoc*)m_pDocument; }
#endif
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

// stdafx.cpp : source file that includes just the standard includes
// stdafx.pch will be the pre-compiled header
// stdafx.obj will contain the pre-compiled type information

#include "stdafx.h"

// stdafx.h : include file for standard system include files,
// or project specific include files that are used frequently, but
// are changed infrequently

#include <afxwin.h> // MFC core and standard components
#include <afxext.h> // MFC extensions (including VB)
#include <afxdb.h> // MFC database classes

//
// ENDOINP.RC2 - resources App Studio does not edit directly
//

#ifdef APSTUDIO_INVOKED
    #error this file is not editable by App Studio
#endif //APSTUDIO_INVOKED

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// Version stamp for this .EXE

#include "ver.h"

VS_VERSION_INFO VERSIONINFO
    FILEVERSION 1,0,0,1
    PRODUCTVERSION 1,0,0,1
    FILEFLAGSMASK VS_FFI_FILEFLAGSMASK
    #ifdef _DEBUG
        FILEETAGS
    VS_FF_DEBUG|VS_FF_PRIVATEBUILD|VS_FF_PRERELEASE
    #else
        FILEFLAGS 0 // final version
    #endif

```

```

FILEOS                VOS_DOS_WINDOWS16
FILETYPE              VFT_APP
FILESUBTYPE           0 // not used
BEGIN
  BLOCK "StringFileInfo"
  BEGIN
    BLOCK "040904E4" // Lang=US English, CharSet=Windows Multilingual
    BEGIN
      VALUE "CompanyName",          "\0"
      VALUE "FileDescription",      "ENDOINP MFC Application\0"
      VALUE "FileVersion",          "1.0.001\0",
      VALUE "InternalName",         "ENDOINP\0"
      VALUE "LegalCopyright",       "\0"
      VALUE "LegalTrademarks",      "\0"
      VALUE "OriginalFilename",     "ENDOINP.EXE\0"
      VALUE "ProductName",          "ENDOINP\0"
      VALUE "ProductVersion",       "111.0.001\01",
    END
  END
END
BLOCK "VarFileInfo"
BEGIN
  VALUE "Translation", 0x409, 1252
  // English language (0x409) and the Windows ANSI codepage
  (1252)
  END
END
////////////////////////////////////
////////////////////////////////////
// Add additional manually edited resources here...

////////////////////////////////////
////////////////////////////////////

//{{NO DEPENDENCIES}}
App Studio generated include file.
//Used by PTDINP.RC

#define APS_3D_CONTROLS      1
#define YDD_TBOUTBOX         100
#define IDD_ENDOIN_FORM      101
#define IDD_ENDO_PG01        102
#define IDD_ENDO_PG02        103
#define IDD_ENDO_PG03        104
#define IDD_ENDO_PG04        105
#define IDD_ENDO_PG05        106
#define IDD_ENDO_PG06        107
#define IDD_ENDO_PG07        108
#define IDD_ENDO_PG08        109
#define IDD_ENDO_PG09        110
#define IDD_ENDO_PG10        111
#define IDD_ENDO_PG11        112
#define IDD_ENDO_PG12        113
#define IDD_ENDO_PG13        114
#define IDD_ENDO_PG14        115
#define IDD_ENDO_SP04A        116
#define IDD_ENDO_SP04B        117
#define IDD_ENDO_SP07A        118
#define IDD_ENDO_SP08A        119
#define IDD_ENDO_SP08B        120

```

```

#define IDR_MAINFRAME 128
#define IDR_ENDOINTYPE 129
#define IDP_FAILED_OPEN_DATABASE 130
#define IDD_ENDO_S908C 131
#define IDD_ENDO_PG15 132
#define IDD_ENDO_SP10A 133
#define IDD_ENDO_SP08D 134
#define IDD_ENDO_SP09A 135
#define IDD_ENDO_SP08E 136
#define IDD_ENDO_PGO 137
#define IDD_ENDO_PG77 138
#define IDD_D_PT5_INP 139
#define IDD_D_GOT5 140
#define IDB_BITMAP1 141
#define IDC_E_DATE 1000
#define IDC_E_ADEZA_ID 1001
#define IDC_E_INST_ID 1002
#define IDC_E_AGE_MENS2 1002
#define IDC_E_ADEZA_ID2 1002
#define IDC_E_TOTAL_POINTS 1002
#define IDC_E_PAT_BIRTHDATE 1003
#define IDC_E_PAT_ZIPCODE 1004
#define IDC_E_PAT_OCCUPATION 1005
#define IDC_C_PAT_WHITE 1006
#define IDC_C_PAT_BLACK 1007
#define IDC_C_PAT_HISPANIC 1008
#define IDC_C_PAT_ASIAN 1009
#define IDC_C_PAT_OTHER 1010
#define IDC_C_PAT_HIGHSCHOOL 1011
#define IDC_C_PAT_COLLEGE 1012
#define IDC_C_PAT_GRADUATE 1013
#define IDC_C_PAT_POSTGRAD 1014
#define IDC_C_MARRIED 1015
#define IDC_B_GOBACK 1016
#define IDC_C_SP_WHITE 1017
#define IDC_C_SP_BLACK 1018
#define IDC_C_SP_HISPANIC 1019
#define IDC_C_SP_ASIAN 1020
#define IDC_C_SP_OTHER 1021
#define IDC_C_SP_HIGHSCHOOL 1022
#define IDC_C_SP_COLLEGE 1023
#define IDC_C_SP_GRADUATE 1024
#define IDC_C_SP_POSTGRAD 1025
#define IDC_E_SP_OCCUPATION 1026
#define IDC_E_PAT_AGE 1027
#define IDC_C_PAT_FLAG 1028
#define IDC_R_DIAB_MELL1 1029
#define IDC_R_DIAB_MELL2 1030
#define IDC_R_DIM_MELL3 1031
#define IDC_B_PREV_PG 1032
#define IDC_R_OTHER_STD1 1033
#define IDC_R_OTHER_STD2 1034
#define IDC_R_PI_DIAB1 1035
#define IDC_R_PI_DIAB2 1036
#define IDC_R_PI_DIAB3 1037
#define IDC_R_OTHER_STD3 1038
#define IDC_R_VAG_IRF1 1039
#define IDC_R_VAG_INF2 1040
#define IDC_R_VAG_INF3 1041
#define IDC_R_GEN_WARTS1 1042
#define IDC_R_GEN_WARTS2 1043

```

```

#define IDC_R_GEN_WARTS3 1044
#define IDC_R_UT_YUB_ABNORI 1045
#define IDC_R_UT_TUB_ABNOR2 1046
#define IDC_R_UT_TUB_ABNOR3 1047
#define IDC_R_DYS_UT_BLEED1 1048
#define IDC_R_DYS_UT_BLEED2 1049
#define IDC_R_HYPERTEN1 1050
#define IDC_R_DYS_UT_BLEED3 1050
#define IDC_R_HYPERTEN2 1051
#define IDC_R_SMOKING1 1051
#define IDC_R_HYPERTEN3 1052
#define IDC_R_SMOKING2 1052
#define IDC_R_PI_HYPERTEN1 1053
#define IDC_R_S140_KING3 1053
#define IDC_R_PI_HYPERTEN2 1054
#define IDC_R_DRUG_ABUSE1 1054
#define IDC_R_PI_HYPERTEN3 1055
#define IDC_R_DRUG_ABUSE2 1055
#define IDC_R_AUTO_IMMUNE1 1056
#define IDC_R_DRUG_ABUSE3 1056
#define IDC_R_AUTO_IMMUNE2 1057
#define IDC_R_PRES_MED1 1057
#define IDC_R_AUTO_IMMUNE3 1058
#define IDC_R_PRES_MED2 1058
#define IDC_R_PRES_MED3 1059
#define IDC_R_DYS_UT_BLEED4 1059
#define IDC_R_UNDETERMINED2 1060
#define IDC_R_OV_CYST1 1061
#define IDC_R_ORG_TRANS1 1062
#define IDC_R_OV_EYST2 1062
#define IDC_R_ORG_TRANS2 1063
#define IDC_R_OV_CYST3 1063
#define IDC_R_OTHER_CUR1 1063
#define IDC_R_ORG_TRANS3 1064
#define IDC_R_POLY_OV_SYNDI 1064
#define IDC_R_OTHEK_CUR2 1064
#define IDC_R_PEL_INFL_DIS1 1065
#define IDC_R_POLY_OV_SYND2 1065
#define IDC_R_PEL_INFL_DIS2 1066
#define IDC_R_POLY_OV_SYND3 1066
#define IDC_R_PEL_INFL_DIS3 1067
#define IDC_R_AB_PAP_DYSPL1 1067
#define IDC_R_HERPEST1 1068
#define IDC_R_AB_PAP_DYSPL2 1068
#define IDC_R_HERPES2 1069
#define IDC_R_AB_PAP_DYSPL3 1069
#define IDC_R_HERPES3 1070
#define IDC_R_GYN_CANSE3 1070
#define IDC_R_GYN_CANSE2 1071
#define IDC_R_GYN_CANSE1 1072
#define IDC_R_FIBTOIDS3 1073
#define IDC_R_FIBROIDS2 1074
#define IDC_R_FIBROIDS1 1075
#define IDC_R_OTHER_HX3 1076
#define IDC_R_OTHER_HX2 1077
#define IDC_R_OTHER_HX1 1078
#define IDC_R_PELVIE_PAIN1 1079
#define IDC_R_ECTOPIC_PREG1 1079
#define IDC_R_PELVIC_PAIN2 1080
#define IDC_R_ECTOPIC_PREG2 1080
#define IDC_R_ABDOM_PAIN1 1081

```

```

#define IDC_R_ECTOPIC_PREG3 1081
#define IDC_ABDOM_PAIN2 1082
#define IDC_R_MENS_ABNORM1 1083
#define IDC_R_MENS_ABNORM2 1084
#define IDC_R_DYSMEN1 1085
#define IDC_R_DYSMEN2 1086
#define IDC_R_DISPAR1 1087
#define IDC_R_DISPAR2 1088
#define IDC_R_INFERTILITY1 1089
#define IDC_R_INFERTILITY2 1090
#define IDC_R_ADN_MAS_THICK1 1091
#define IDC_R_ADN_MAS_THICK2 1092
#define IDC_R_OVARIAN_CYST1 1093
#define IDC_R_OVARIAN_CYST2 1094
#define IDC_R_UNDETERMINED1 1095
#define IDC_E_CUR_SYM_OTHER 1096
#define IDC_R_MENST_REG1 1097
#define IDC_R_MENST_REG2 1098
#define IDC_E_LAST_PERIOD 1099
#define IDC_E_RECENT_PART 1100
#define IDC_E_GRAVIDITY 1101
#define IDC_E_PARITY 1102
#define IDC_R_HX_INFERT1 1103
#define IDC_R_HX_INFERT2 1104
#define IDC_R_OV_STAT_KNOWN1 1105
#define IDC_R_OV_STAT_KNOWN2 1106
#define IDC_E_SPONT_ABORT 1107
#define IDC_R_MENS_FLOW1 1107
#define IDC_E_ELECT_ABORT 1108
#define IDC_R_MENS_FLOW2 1108
#define IDC_R_MENS_FLOW3 1109
#define IDC_R_HX_OF_END01 1110
#define IDC_R_HX_OF_END= 1111
#define IDC_R_HX_PEL_SURG1 1112
#define IDC_R_HX_PEL_SURG2 1113
#define IDC_R_HORMON_MED1 1114
#define IDC_R_HORMONE_MED2 1115
#define IDC_E_CUR_SURG_DATE 1116
#define IDC_E_CUR_SURG_REASON2 1117
#define IDC_C_DIAG_LA_PAR 1118
#define IDC_CLASER_OBLIT 1119
#define IDC_C_SURG_EXCISION 1120
#define IDC_C_BI_SAL_OOPH 1121
#define IDC_C_UNIL_OOPH 1122
#define IDC_C_EXC_OV_CYST 1123
#define IDC_C_OLULA 1124
#define IDC_C_HYSTERECTOMY 1125
#define IDC_C_HYSTEROSCOPY 1126
#define IDC_C_D_AND_C 1127
#define IDC_C_CUR_SURG_OTHER 1128
#define IDC_C_NORM_PEL 1129
#define IDC_ENDO_PRESENT 1130
#define IDC_C_ADHESIONS_PRES 1131
#define IDC_C_FIBROIDS_TRES 1132
#define IDC_C_PELV_INF_DISEASE 1133
#define IDC_C_GYN_CANCER 1134
#define IDC_C_OTHER_GYN_DIS 1135
#define IDC_R_AFS_STG1 1136
#define IDC_R_AFS_STG2 1137
#define IDC_R_AFS_STG3 1138
#define IDC_R_AFS_STG4 1139

```

```

#define IDC_C_BLUE_BK_LESIONS 1141
#define IDC_C_RED_LESIONS 1142
#define IDC_C_WHITE_LESIONS 1143
#define IDC_R_PEL_ADH_PRES1 1144
#define IDC_R_PEL_ADH_PRES2 1145
#define IDC_R_AW_CORF_BIOPSY1 1146
#define IDC_R_ENDO_CONF_BIOPSY2 1147
#define IDC_b-OVERIES_AT 1148
#define IDC_C_OVERIES_ADH 1149
#define IDC_C_FALLOP_EST 1150
#define IDC_C_FALLOP_ADH 1151
#define IDC_C_UT_LIG_EST 1152
#define IDC_C_UT_LIG_ADH 1153
#define IDC_C_CULDESAC_EST 1154
#define IDC_C_CULDESAC_ADH 1155
#define IDC_C_BROAD_LIG_EST 1156
#define IDC_C_BROAD_LIG_ADH 1157
#define IDC_C_PEL_SYDE_EST 1158
#define IDC_C_PEL_SIDE_ADH 1159
#define IDC_C_VESIC_EST 1160
#define IDC_C_VESIC_ADH 1161
#define IDC_C_OTHER_EST 1162
#define IDC_C_OTHER_ADH 1163
#define IDC_E_PID_DATE 1164
#define IDC_R_HAVE_PID1 1165
#define IDC_E_PID_LOC_SPECIFY 1165
#define IDC_E_ADDL_PID 1165
#define IDC_R_HAVE_PID2 1166
#define IDC_E_PID_LOC_SPECIFY2 1166
#define IDC_C_PID_C_LAPS 1167
#define IDC_C_PID_C_LAPT 1168
#define IDC_C_PID_C_DIFF_DIAG 1169
#define IDC_C_PID_C_UNDET 1170
#define IDC_E_PID_A_SPECIFY 1171
#define IDC_R_PID_CONF_SURG1 1172
#define IDC_R_GC_HISTOLOGY1 1172
#define IDC_R_P15_CONF_SURG2 1173
#define IDC_R_GC_HISTOLOGY2 1173
#define IDC_C_PID_M_ORG_UNKNOWN 1174
#define IDC_C_PID_ORG_NEISS 1175
#define IDC_C_PID_M_ORG_CH_TR 1176
#define IDC_C_PID_M_ORG_GM 1177
#define IDC_C_PID_M_ORG_OTHER 1178
#define IDC_C_PID_M_LOC_VAGINA 1179
#define IDC_C_PID_LOC_CERVIX 1180
#define IDC_C_PID_LOC_OVARIES 1181
#define IDC_C_PID_LOC_FALLOP 1182
#define IDC_C_PID_LOC_OTHER 1183
#define IDC_E_PID_ORG_SPECIFY 1184
#define IDC_R_GC_PRIMARY 1185
Nefine IDC_R_GC_PRIMARY2 1186
Nefine IDC_R_GC_PRIMARY3 1187
#define IDC_R_GC_PRIMARY4 1188
#define IDC_R_GC_GRADE1 1189
Kefine IDC_R_GC_STAGE2 1190
Kefine IDC_R_GC_STAGE3 1191
Nefine IDC_R_GC_STAGE4 1192
#define IDC_R_GC_STAGE5 1193
#define IDC_R_GC_GRADE2 1194
#define IDC_R_GC_GRADE3 1195
#define IDC_R_GC_STAGE1 1196

```

```

#define IDC_E_GC_TUMOR_TYPE 1197
#define IDC_E_GC_SITES_SPECIFY 1198
#define IDC_E_GC_ADD_INFO 1199
#define IDC_E_PKS_PER_DAY 1200
#define IDC_E_OTHER_STD_SPECIFY 1201
#define IDC_E - PRES-MED DRUG1 1202
#define IDC_E PRES_MED_DATE1 1203
#define IDC_E PRES_MED_DRUG2 1204
#define IDC_E_OTHER_HX_SPECIFY 1204
#define IDC_E PRES_MED_DATE2 1205
#define IDC_C INFERT PRI 1205
#define IDC_E PRIMARY_LEN 1206
#define IDC_C INFERT_SEC 1207
#define IDC_C HOR_MED1 1207
#define IDC_ESECONDARY_LEN 1208
#define IDC_E HOR_MED_DOSE1 1208
#define IDC_E HOR_MED_DATE1 1209
#define IDC_E PEL_SURG_TYPE1 1209
#define IDC_E HOR_MED_PURP1 1210
#define IDC_E PEL_SURG_DATE1 1210
#define IDC_C HOR_MED2 1211
#define IDC_E PEL_SURG_DATE2 1211
#define IDC_R OVUL_STAT1 1211
#define IDC_E HOR_MED_DOSE2 1212
#define IDC_E PEL_SURG_TYPE2 1212
#define IDC_R OVUL_STAT2 1212
#define IDC_E HOR_MED_DATE2 1213
#define IDC_E PEL_SUR_DATE3 1213
#define IDC_R OVUL_STAT3 1213
#define IDC_E HOR_MED_PURP2 1214
#define IDC_E PEL_SURG_TYPE3 1214
#define IDC_EST_OTHER_SPECIFY 1214
#define IDC_C HOR_MED 1215
#define IDC_E PEL_SURG_DATE4 1215
#define IDC_E ADH-OTHER_SPECIFY 1215
#define IDC_E HOR_MED_DOSE3 1216
#define IDC_E PEL_SURG_TYPE4 1216
#define IDC_C MENST_HORM_INDUCED 1216
#define IDC_E HOR_MED_DATE3 1217
#define IDC_E TYP_CYC_LEN 1217
#define IDC_E HOR_MED_PURP3 1218
#define IDC_E TYP_PERIOD_LEN 1218
#define IDC_C HOR_MED4 1219
#define IDC_E FREQUENCY 1219
#define IDC_E HOR_MED_DOSE4 1220
#define IDC_E_OTH_SURG_PROC_SPECIFY 1220
#define IDC_E HOR_MED_DATE4 1221
#define IDC_E_OTHER_GYN_SPECIFY 1221
#define IDC_E HOR_MED_PURP4 1222
#define IDC_CONFIRMED_BY_LAPAROSCOPY 1222
#define IDC_C_CONFIRMED_BY_LAPAROTOMY 1223
#define IDC_CONFIRMED_BY_BIOPSY 1224
#define IDC_E_LAPAROSCOPY_DATE 1225
#define IDC_E_LAPAROTOMY_DATE 1226
#define IDC_E_BIOPSY_DATE 1227
#define IDC_E_RECORD_COUNT 1230
#define IDC_R_HORMONE_INDUCED 1232
#define IDC_R_HORMONE_INDUCED2 1233
#define IDC_EO_WHITE 1247
#define IDC_EO_BLACK 1248
#define IDC_EO_ASIAN 1249

```

```

#define IDC_EO_HISPANIC 1250
#define IDC_EO_NATIVE_AMERICAN 1251
#define IDC_EO_OTHER 1252
#define IDC_MS_MARRIED 1253
#define IDC_MS_SINGLE 1254
#define IDC_MS_WIDOWED 1255
#define IDC_MS_LWP 1256
#define IDC_MS_OTHER 1257
#define IDC_ACOG_Y 1258
#define IDC_ACOG_N 1259
#define IDC_MS_DIVORCED 1260
#define IDC_ANTIBIOTICS 1261
#define IDC_FFN_POS 1262
#define IDC_CORTICOSTEROIDS 1263
#define IDC_TOCOLYTICS 1264
#define IDC_INSULIN 1265
#define IDC_ANTIHYPER 1266
#define IDC_FFN_NEG 1267
#define IDC_MED_NONE 1268
#define IDC_MED_UKN 1269
#define IDC_PATENT_COMP_1 1270
#define IDC_PATIENT_COMP_3 1271
#define IDC_PC1_LT1 1272
#define IDC_PC1_1_3 1273
#define IDC_PC1_4_6 1274
#define IDC_PATENT_COMP_2 1275
#define IDC_VAGINAL_BLEEEING 1276
#define IDC_VB_TRAC 1277
#define IDC_VB_MED 1278
#define IDC_VB_GROSS 1279
#define IDC_PATIENT_COMP_6 1280
#define IDC_PATIENT_COMP_5 1281
#define IDC_PATIENT_COMP_4 1282
#define IDC_EGA_BY_SONO 1283
#define IDC_EGA_BY_LMP 1284
#define IDC_EGA_AT_SAMP 1285
#define IDC_DILITATION_LT1 1286
#define IDC_DILITATION_1 1287
#define IDC_DILITATION_1_2 1288
#define IDC_DILITATION_2 1289
#define IDC_DILITATION_2_3 1290
#define IDC_DILITATION_3 1291
#define IDC_DILITATION_GT3 1292
#define IDC_CERV_FIRM 1293
#define IDC_CERV_MOD 1294
#define IDC_CERV_SOFT 1295
#define IDC_1_CORP 1298
#define IDC_2_COMP 1299
#define IDC_3_COMP 1300
#define IDC_4_COMP 1301
#define IDC_5_COMP 1302
#define IDC_6_COMP 1303
#define IDC_MULT_GEST 1304
#define IDC_UT_CWRV_ABNORM 1305
#define IDC_CERV_CERCLAGE 1306
#define IDC_GEST_DIABETES 1307
#define IDC_HYPERTEN_DISORDERS 1308
#define IDC_MG_TWINS 1309
#define IDC_MG_TRIPLETS 1310
#define IDC_MG_QUADS 1311
#define IDC_NAME_L 1313

```

```

#define IDC_NAME_F 1314
#define IDC_NAME_MI 1315
#define IDC_DATE_OF_BIRTH 1316
#define IDC_LAB_ID 1317
#define IDC_DILATATION_UKN 1318
#define IDC_GRAVIDITY 1319
#define IDC_PARITY 1320
#define IDC_ABORTIONS 1321
#define IDC_PC1_7_9 1322
#define IDC_PC1_10_12 1323
#define IDC_PC1_GT12 1324
#define IDC_2_CUMP_1 1325
#define IDC_2_COMP_2 1326
#define IDC_2_COMP_3 1327
#define IDC_R_GOTO_SEL1 1329
#define IDC_R_GOTO_SEL2 1330
#define IDC_E_GOTO_REC_NUM 1331
#define IDC_E_GOTO_ID_NUM 1332
#define IDD_DATA_NEW 32771
#define ID_DATA_NEW 32772
#define ID_DATA_EDIT 32773
#define ID_REC_FIRST 32774
#define ID_REC_NEXT 32775
#define ID_REC_PREV 32776
#define ID_REC_LAST 32777
#define ID_BLD_NET_FILE 32778
#define ID_EDIT_MODE 32779
#define ID_CLR_SUBFIELDS 32780
#define ID_REC_GOTO 32781

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS

#define APS_NEXT_RESOURCE_VALUE 142
#define APS_NEXT_COMMAND_VALUE 32782
#define APS_NEXT_CONTROL_VALUE 1333
#define APS_NEXT_SYMED_VALUE 101
#endif
#endif

```

```

//Microsoft App Studio generated resource script.
//
#include "resource.h"

```

```

#define APSTUDIO_READONLY_SYMBOLS
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "afxres.h"
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//
#undef APSTUDIO_READONLY_SYMBOLS
#ifdef APSTUDIO_INVOKED

```

```

////////////////////////////////////
////////////////////////////////////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE DISCARDABLE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
    "#include "'afxres.h'\r\n"
    "\0"
END

3 TEXTINCLUDE DISCARDABLE
BEGIN
    #include "res\\PTDinp.rc2" // non-App Studio edited
resources\r\n"
    "\r\n"
    "#include "'afxres.rc" \011// Standard components \r\n"
    "#include "'afxprint.rc" \011// printing/print preview
resources\r\n"
    "#include "afxdb.rc"\011\011// Database resources\r\n"
    "\0"
END
////////////////////////////////////
////////////////////////////////////
#endif //APSTUDIO_INVOKED

////////////////////////////////////
////////////////////////////////////
//
// Icon
//
IDR_MAINFRAME ICON DISCARDABLE "RES\\PTDINP.ICO"

////////////////////////////////////
////////////////////////////////////
//
// Bitmap
//

IDR_MAINFRAME BITMAP MOVEABLE PURE "RES\\TOOLBAR.BMP11
IDB_BITMAP1 BITMAP DISCARDABLE
"RES\\BITMA.P1.BMP"

////////////////////////////////////
////////////////////////////////////
//
// Menu
//
IDR_MAINFRAME MENU PRELOAD DISCARDABLE

BEGIN
    POPUP "&File"
    BEGIN
        MENUITEM " &Open...\tCtrl+O", ID_FILE_OPEN
        MENUITEM SEPARATOR
        MENUITEM "&Print", ID_FILE_PRINT
    
```

```

MENUITEM "Print &Setup".          ID_FILE_PRINT_SETUP
MENUITEM "Print Preview",         ID_FILE_PRINT_PREVIEW
MENUITEM SEPARATOR
MENUITEM "File1",                 ID_FILE_MRU_FILE1, GRAYED
MENUITEM "File2",                 ID_FILE_MRU_FILE2, GRAYED
MENUITEM "File3",                 ID_FILE_MRU_FILE3, GRAYED
MENUITEM "File4",                 ID_FILE_MRU_FILE4, GRAYED
MENUITEM SEPARATOR
MENUITEM "E&xit",                  ID_APP_EXIT

END
POPUP "&Record"
BEGIN
    MENUITEM "%First Record",      ID_REC_FIRST
    MENUITEM "&Prev Record",        ID_REC_PREV
    MENUITEM "%Next Record",       ID_REC_NEXT
    MENUITEM "%Last Record",       ID_REC_LAST
    MENUITEM SEPARATOR
    MENUITEM "%Go to Record",      ID_REC_GOTO
    MENUITEM SEPARATOR
    MENUITEM "%Edit Record",       ID_DATA_EDIT
    MENUITEM "%New Record",        ID_DATA_NEW
    MENUITEM SEPARATOR
    MENUITEM "Neural &Data",        ID_BLD_NET_FILE

END
POPUP "%Options"
BEGIN
    MENUITEM "%Print Full Form",    ID_EDIT_MODE
    MENUITEM "%Clear Subfields",    ID_CLR_SUBFIELDS

END
POPUP "%View"
BEGIN
    MENUITEM "%Toolbar".           ID_VIEW_TOOLBAR
    MENUITEM "%Status Bar",        ID_VIEW_STATUS_BAR

END
POPUP "%Help"
BEGIN
    MENUITEM "%About PTDinp",       ID_APP_ABOUT

END

END

////////////////////////////////////
////////////////////////////////////
//
// Accelerator
//
IDR_MAINFRAME ACCELERATORS PRELOAD MOVEABLE PURE
BEGIN
    "N"          ID_FILE_NEW,          VIRTKEY, CONTROL
    "O"          ID_FILE_OPEN,         VIRTKEY, CONTROL
    "S"          ID_FILE_SAVE,         VIRTKEY, CONTROL
    "P"          ID_FILE_PRINT,        VIRTKEY, CONTROL
    "Z"          ID_EDIT_UNDO,         VIRTKEY, CONTROL
    "X",         ID_EDIT_CUT,          VIRTKEY, CONTROL
    "C"          ID_EDIT_COPY,         VIRTKEY, CONTROL
    "V"          ID_EDIT_PASTE,        VIRTKEY, CONTROL
    VK_BACK,     ID_EDIT_UNDO,         VIRTKEY, ALT
    VK_DELETE,   ID_EDIT_CUT,          VIRTKEY, SHIFT
    VK_INSERT,   ID_EDIT_COPY,         VIRTKEY, CONTROL
    VK_INSERT,   ID_EDIT_PASTE,        VIRTKEY, SHIFT
    V1_F6        ID_NEXT_PANE,        VIRTKEY

```

```

VK_F6, ID_PREV_PANE, VIRTKEY,SHIFT
END

////////////////////////////////////
////////////////////////////////////
//
//Dialog
//

IDD_ABOUTBOX DIALOG DISCARDABLE 34, 22, 217, 55
STYLE DS_MODALFRAME I | WS_POPUP | WS_CAPTION | I WS_SYSMENU
CAPTION "About PTDinp"
FONT 8, "MS Sans Serif"
BEGIN
    ICON IDR_MAINFRAME, IDC_STATIC, 11, 17, 18, 20
    LTEXT "Pre Term Delivery Application Version 1.0",
    IDC_STATIC, 40,10,139,8
    LTEXT "Copyright \251 1997 ",IDC_STATIC, 40,25,119,8
    DEFPUSHBUTTON "OK", IDOK, 175, 32, 32, 14,WS_GROUP
END

IDD_D_PTD_INP DIALOG DISCARDABLE 0, 0, 399, 447
STYLE-DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION |
WS_SYSMENU
CAPTION "Pre-Term Delivery Risk Assessment Software: Data Entry Screen"
FONT 8, "MS Sans Serif"
BEGIN
    EDITTEXT IDC_LAB_ID,305,8,68,12,ES_AUTOHSCROLL
    EDITTEXT IDC_NAME_L, 46, 48, 50, 13,
    ES_AUTOHSCROLL
    EDITTEXT IDC_NAME_F, 117, 48, 40, 13,
    ES_AUTOHSCROLL
    EDITTEXT IDC_NAME_MI,170,48,12,13,ES_AUTOHSCROLL
    EDITTEXT IDC_DATE_OF_BIRTH, 28, 66, 59, 12,
    ES_AUTOHSCROLL
    CONTROL "Caucasian", IDC_EO_WHITE, "Button"
    BS_AUTOCHECKBOX |
    WS_TABSTOP,242, 48,45, 10
    CONTROL "African American", IDC_EO_BLACK,
    "Button", BS_AUTOCHECKBOX |
    WS_TABSTOP,292,48,66,1
    CONTROL "Asian", IDC_EO_ASIAN, "Button",
    BS_AUTOCHECKBOX |
    WS_TABSTOP, 362, 48,29,10
    CONTROL "Hispanic", IDC_EO_HISPANIC,
    "Button",BS_AUTOCHECKBOX |
    WS_TABSTOP,242,59,40,10
    CONTROL "Native American", IDC_EO_NATIVE_AMER I
    CAN, "Button",
    BS_AUTOCHECKBOX |
    WS_TABSTOP,292,59,65,10
    CONTROL "Other ", IDC_EO_OTHER, "Button",
    BS_AUTOCHECKBOX |
    WS_TABSTOP,362,59,29,10
    CONTROL "Married",IDC_MS_MARRIED, "Button",
    BS_AUTOCHECKBOX |
    WS_TABSTOP,242, 72,36,10
    CONTROL "Single",IDC_I_MS_SINGLE, "Button",
    BS_AUTOCHECKBOX |
    WS_TABSTOP,262,72,34,10

```

CONTROL  
 "Divorced/Separated", IDC\_MS\_DIVORCED, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 316, 72, 77, 10  
 CONTROL "Widowed ", IDC\_MS\_WIDOWED, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 242, 83, 41, 10  
 CONTROL "Living with partner", IDC\_MS\_LWP,  
 "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 287, 83, 73, 10  
 CONTROL "Other", IDC\_MS\_OTHER,  
 "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP, 562, 83, 29, 10  
 CONTROL "Yes", IDC\_ACOG\_Y, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 333, 119, 24, 10 -  
 CONTROL "No", IDC\_ACOG\_N, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 364, 119, 21, 10  
 CONTROL "Uterine contractions with or without  
 pain",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 13, 145, 143, 10  
 CONTROL " <1", IDC\_PC1\_LT1, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 67, 158, 20, 10  
 CONTROL "1-3", IDC\_PC1\_1\_3, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 99, 158, 22, 10  
 CONTROL "4-6", IDC\_PC1\_4\_6, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 131, 158, 22, 10  
 CONTROL "7-9", IDC\_PC1\_7\_9, "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP, 67, 170, 22, 10  
 CONTROL "10-12" Button", BS\_AUTOCHECKBOX |  
 WS\_TABSTOP, 99, 170, 30, 10  
 CONTROL ">12", IDC\_PC1\_GT12, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 131, 170, 24, 10  
 CONTROL "Vaginal bleeding",  
 IDC\_VAGINAL\_BLEEDING, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 13, 181, 65, 10  
 CONTROL "Trace", IDC\_VB\_TRACE, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 23, 194, 30, 10  
 CONTROL "Med",  
 IDC\_VB\_MED, "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP, 58, 194, 25, 10  
 CONTROL "Gross", IDC\_VB\_GROSS, "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP 88, 194, 30, 10  
 CONTROL "Patient is not ""feeling  
 right""", IDC\_PATIENT\_COMP\_6,  
 "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP, 13, 205, 102, 10  
 CONTROL "Bleeding during the second or third  
 trimester",  
 IDC\_PATIENT\_COMP\_3,  
 Button", BS\_AUTOCHECKBOX | WS\_TABSTOP, 161 145, 155 10  
 CONTROL "Intermittent lower abdominal pain,  
 dull, low backpain, pelvic press ure",  
 IDC\_PATIENT\_COMP\_2, "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP, 161, 157, 233, 10  
 CONTROL "Change in vaginal discharge - -  
 amount, color, or consistency",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 161 181, 208 10  
 CONTROL "Menstrual - like cramping (with or  
 without diarrhea)",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 161 193, 171 10  
 IDC\_PATIENT\_COMP\_4, "Button",

EDITTEXT IDC\_EGA\_BY\_SONO, 155,224,37,12,  
 ES\_AUTOHSCROLL  
 EDITTEXT IDC\_EGA\_BY\_LMP,  
 245,224,37,12,ES\_AUTOHSCROLL  
 EDITTEXT IDC\_EG\_AT\_SAMP,  
 350,224,37,12,ES\_AUTOHSCROLL  
 CONTROL "Previous pregnancy, no  
 complications",IDC\_1\_COMP,  
 "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP,13, 260, 134,10  
 CONTROL "History of Preterm  
 delivery",IDC\_2\_COMP,"Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 13, 272, 134, 10  
 CONTROL "1",  
 IDC\_2\_COMP\_1,"Button",BS\_AUTOCHECKBOX | WS\_TABSTOP, 91, 284, 19, 10 -  
 CONTROL "1", IDC\_2\_COMP\_2,"Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 116, 284 19,10  
 CONTROL ">2", IDC\_2\_COMP\_3,"Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 141,284, 21, 10 -  
 CONTROL "History of Preterm  
 PROM",IDC\_3\_COMP,"Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 13, 296, 92, 10  
 CONTROL "History of incompetent cervix",  
 IDC\_4\_COMP,"Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP,13, 308, 106,10  
 CONTROL "History of PIH/preeclampsia",  
 IDC\_5\_COMP,"Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP,13 , 320, 102, 10  
 CONTROL "History of SAB prior to 20 wks",  
 IDC\_6\_COMP,"Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP,13,332, 109, 10  
 EDITTEXT IDE\_GRAVIDITY, 277, 246,20, 12,  
 ES\_AUTOHSCROLL  
 EDITTEXT IDC\_PARITY, 317, 246, 20, 12  
 ES\_AUTOHSCROLL  
 EDITTEXT IDC\_ABORTIONS, 357, 246, 20, 12,  
 ES\_AUTOHSCROLL  
 CONTROL "Multiple  
 Gestation:",IDC\_MULT\_GEST,"Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 23, 272, 72, 10  
 CONTROL "Twins", IDC\_MG\_TWINS, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 278, 272,30,10  
 CONTROL "Triplets", IDC\_MG\_TRIPLETS, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 311, 272, 36, 10  
 CONTROL "Quads", IDC\_MG\_QUADS, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP,550, 272, 32, 10  
 CONTROL "Uterine or cervical abnormality",  
 IDC\_UT\_CWRV\_ABNORM, "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP, 203,  
 284,110,10  
 CONTROL "Cerclage", TDC\_CERV\_CERCLAGE,  
 "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP,203, 296, 40, 10  
 CONTROL "Gestational  
 Diabetes",IDC\_GEST\_DIABETES, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 203, 308, 79, 10  
 CONTROL "Hypertensive Disorders",  
 IDC\_HYPERTEN\_DISORDERS, "Button", BS\_AUTOCHECKBOX | WS\_TABSTOP, 203, 320,  
 86, 10  
 CONTROL "1", IDC\_DILITATION\_LT1, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 58, 364, 22, 10  
 CONTROL "1", IDC\_DILITATION\_1, "Button",  
 BS\_AUTOCHECKBOX | WS\_TABSTOP, 81, 364, 24, 10

CONTROL		"1-2", IDC_DILITATION_1_2, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP 101,	364, 24, 10
CONTROL		"2", IDC_DILITATION_2, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 127,	364, 18, 10
CONTROL		"2-3", IDC_DILITATION_2_3, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP 147,	364, 24, 10
CONTROL		"3", IDC_DILITATION_3, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 173,	364, 18, 10
CONTROL		">3", IDC_DILITATION_GT3, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 193,	364, 22, 10
CONTROL		"Unk. ", IDC_DILITATION_UKU, "Button",
BS_AUTOCHECKBOX	WS TABSTOP, 217,	364, 29, 10
CONTROL		Firm", IDC_CERV_FIRM, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 318,	564, 25, 10
CONTROL		"Mod" IDC_CERV_MOD, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 344,	364, 25, 10
CONTROL		"Soft", IDC_CERV_SOFT, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 370,	64, 25, 10
CONTROL		"Antibiotics", IDC_ANTIBIOTICS, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 17,	392, 45, 10
CONTROL		"Corticosteroids", IDC_CORTICOSTEROIDS,
"Button",		
BS_AUTOCHECKBOX	WS_TABSTOP, 70,	392, 60, 10
CONTROL		"Tocolytis", IDC_TOCOLYTICS, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 138,	592, 41, 10
CONTROL		"Insulin", IDC_INSULIN, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 187,	392, 33, 10
CONTROL		"Antihypertensive ", IDC_ANTIHYPER,
"Button",		
BS_AUTOCHECKBOX	WS_TAESTOP, 228,	392, 69, 10
CONTROL		"None", IDC_MED_NONE, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 305,	392, 29, 10
CONTROL		"Unknown", IDC_MED_UKN, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 342,	392, 42, 10
CONTROL		"Positive", IDC_FFN_POS, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 138,	411, 37, 10
CONTROL		"Negative", IDC_FFN_NEG, "Button",
BS_AUTOCHECKBOX	WS_TABSTOP, 228,	411, 41, 10
DEFPUSHBUTTON		"Calculate Risk", IDOK, 270, 429, 62, 14
PUSHBUTTON		"Cancel". IDCANCEL, 340, 429, 53, 14
LTEXT		"Cervical consistancy", IDC_STATIC,
249, 365, 68, 8		
LTEXT		"M", IDC_STATIC, 160, 51, 7, 8
LTEXT		"Lab ID #:11, IDC_STATIC, 267, 10, 34,
8		
LTEXT		"PATIENT INFORMATION", IDC_STATIC, 159,
29, 83, 8		
LTEXT		"Name (last) ", IDC_STATIC, 7, 51, 36,
8		
LTEXT		"First", IDC_STATIC, .99, 51, 15, 8
GROUPBOX		"", IDC_STATIC, 1, 40, 187, 56
GROUPBOX		"", IDC_STATIC, 187, 40, 210, 56
LTEXT		"Ethnic origin: ", IDC_STATIC, 192, 48,
44, 8		
LTEXT		"Marital
status:", IDC_STATIC, 192, 72, 47, 8		
LTEXT		"DOB", IDC_STATIC, 7, 69, 16, 8
LTEXT		"PATIENT HISTORY AND CLINICAL
INFORMATION", IDC_STATIC, 117, 102, 168, 8		
GROUPBOX		"", IDC_STATIC, 1, 112, 396, 107

```

LTEXT
patient experiencing signs and symptoms of possible preterm labor?",
IDC_STATIC,7,119,321,8
LTEXT
"If yes, please mark all that apply.
",IDC_STATIC,1,373,396,32
IDC_STATIC,7,134,109,8
GROUPBOX
LTEXT
"Qualitative fFN Elisa Test Results:
IDC_STATIC,1,402,396,24
IDC_STATIC,7,411,118,8
GROUPBOX
LTEXT
"Medications at Time of Test (check all
that apply) ",
IDC_STATIC,7,380,163,8
LTEXT
"Number/hr", IDC_STATIC,22,158,36,8
GROUPBOX
" ",IDC_STATIC,1,216,396,25
LTEXT
"Gestational Age: EGA by first
IDC_STATIC,7,225,143,8
trimester sono",
"EGA by LMP", IDC_STATIC,197,225,42,
8
LTEXT
"EGA at sampling",
LTEXT
" ",IDC_STATIC,1,346,396,30
IDC_STATIC,287,225,55,8
GROUPBOX
LTEXT
"Cervical Status immediately following
sample collection:",
IDC_STATIC,7,352,182,8
LTEXT
"Dilatation (cm)", IDC_STATIC,
9,364,48,8
" ",IDC_STATIC,1,238,187,111
GROUPBOX
" ",IDC_STATIC,187,238,210,111
GROUPBOX
"Previous Pregnancy: Please mark
CONTROL
all that apply.",
IDC_STATIC,"Static",SS_LEFTNOWORDWRAP
| WS_GROUP,7,249,159,8
LTEXT
"Current Pregnancy:
G:",IDC_STATIC,195,249,76,8
" ",IDC_STATIC,1,93,396,22
GROUPBOX
" ",IDC_STATIC,1,1,396,22
GROUPBOX
" ",IDC_STATIC,1,20,396,23
GROUPBOX
"P: ", IDC_STATIC,303,249,8,8
LTEXT
"A: ", IDC_STATIC,343,249,8,8
LTEXT
"If Yes, how many?", IDC_STATIC,22,
284,61,8
END

IDD_D_GOTO DIALOG DISCARDABLE 0, 0, 163, 95
STYLE_DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION |
WS_SYSMENU CAPTION "Go To Record ... "
FONT 8, "MS Sans Serif"
BEGIN
CONTROL
"Record Number", IDC_R_GOTO_SEL1,
"Button",
BS_AUTORADIOBUTTON | WS_GROUP, 10, 16, 62, 10
CONTROL
"ID Number", IDC_R_GOTO_SEL2,
"Button", BS_AUTORADIOBUTTON,10,40,46,10
IDC_E_GOTO_REC_NUM,
EDITTEXT
IDC_E_GOTO_ID_NUM, 90, 36, 60,
90,12,60,12,ES_AUTOHSCROLL
EDITTEXT
IDC_E_GOTO_ID_NUM, 90, 36, 60,
12, ES_AUTOHSCROLL
DEFPUSHBUTTON
"Ok", IDOK,, 76, 50, 14
PUSHBUTTON
"Cancel", IDCANCEL, 100, 76, 50, 14
END

```

```

////////////////////////////////////
////////////////////////////////////
//
// String Table
//
STRINGTABLE PRELOAD DISCARDABLE
BEGIN
    IDR_MAINFRAME                "PTDinp Windows
Application\nPTDin\nPTDin Docuent\n\n\nPTDin. Document\ nPTDin Document"
END

STRINGTABLE PRELOAD DISCARDABLE
BEGIN
    AFX_IDS_APP_TITLE            "PTDinp Windows Application"
    AFX_IDS_IDLEMESSAGE          "Ready"
END

STRINGTABLE DISCARDABLE
BEGIN
    ID_INDICATOR_EXT             "EXT"
    ID_INDICATOR_CAPS            "CAP"
    ID_INDICATOR_NUM             "NUM"
    ID_INDICATOR_SCRL            "SCRL"
    ID_INDICATOR_OVR             "OVR"
    ID_INDICATOR_REC             "REC"
END

STRINGTABLE DISCARDABLE
BEGIN
    ID_FILE_NEW                  "Create a new document"
    ID_FILE_OPEN                  "Open an existing document"
    ID_FILE_CLOSE                 "Close the active document"
    ID_FILE_SAVE                  "Save the active document"
    ID_FILE_SAVE_AS               "Save the active document with a new
name"
    ID_FILE_PAGE_SETUP            "Change the printing options"
    ID_FILE_PRINT_SETUP           "Change the printer and printing
options"
    ID_FILE_PRINT                 "Print the active document"
    ID_FILE_PRINT_PREVIEW         "Display full pages"
END

STRINGTABLE DISCARDABLE
BEGIN
    ID_APP_ABOUT                  "Display program information,
version number and copyright"
    ID_APP_EXIT                   "Quit the application; prompts to save
documents"
END

STRINGTABLE DISCARDABLE
BEGIN
    ID_FILE_MRU_FILE1            "Open this document"
    ID_FILE_MRU_FILE2            "Open this document"
    ID_FILE_MRU_FILE3            "Open this document"
    ID_FILE_MRU7_FILE4           "Open this document"
END

STRINGTABLE DISCARDABLE
BEGIN
    ID_NEXT_PANE                  "Switch to the next window pane"

```

ID\_PREV\_PANE  
window pane"  
END

STRINGTABLE DISCARDABLE  
BEGIN

ID\_EDIT\_CLEAR  
ID\_EDIT\_CLEAR\_ALL  
ID\_EDIT\_COPY  
the Clipboard"  
ID\_EDIT\_CUT  
Clipboard"

ID\_EDIT\_FIND  
ID\_EDIT\_PASTE  
ID\_EDIT\_REPEAT  
ID\_EDIT\_REPLACE  
text"

ID\_EDIT\_SELECT\_ALL  
ID\_EDIT\_UNDO  
ID\_EDIT\_REDO  
action"  
END

STRINGTABLE DISCARDABLE  
BEGIN

ID\_VIEW\_TOOLBAR  
ID\_VIEW\_STATUS\_BAR  
END

STRINGTABLE DISCARDABLE  
BEGIN

AFX\_IDS\_SCSIZE  
AFX\_IDS\_SCMOVE  
AFX\_IDS\_SCMINIMIZE  
AFX\_IDS\_SCMAXIMIZE  
AFX\_IDS\_SCNEXTWINDOW  
AFX\_IDS\_SCPREVWINDOW  
window"

AFX\_IDS\_SCCLOSE  
save the documents"  
END

STRINGTABLE DISCARDABLE  
BEGIN

AFX\_IDS\_SCRESTORE  
AFX\_IDS\_SCTASKLIST  
END

STRINGTABLE DISCARDABLE  
BEGIN

IDD\_DATA\_NEW  
new record"

ID\_DATA\_NEW  
edit."

ID\_DATA\_EDIT  
record."

ID\_REC\_FIRST  
file."

ID\_REC\_NEXT  
ID\_REC\_PREV  
file."

"Switch back to the previous

"Erase the selection"

"Erase everything"

"Copy the selection and put it on

"Cut the selection and put it on the

"Find the specified text"

"Insert Clipboard contents"

"Repeat the last action"

"Replace specific text with different

"Select the entire document?l

"Undo the last action"

"Redo the previously undone

"Show or hide the toolbar"

"Show or hide the status bar"

"Change the window size"

"Change the window position"

"Reduce the window to an icon"

"Enlarge the window to full size"

"Switch to the next document window"

"Switch to the previous document

"Close the active window and prompts to

"Restore the window to normal size"

"Activate-Task List"

"Starts data entry process for

"Create new record at end of file and

"Edit the cturrently selected

"Go to the first record in the

"Go to the next record in the file."

"Go to the previous record in the

```

        ID_REC_LAST           "Go to the last record in the file."
        ID_BID_NET_FILE       "Build file of neural data from
currently opened database."
        ID_EDIT_MODE           "Print the full data form when
checked or results only when unchecked."
        ID_CLR_SUBFIELDS       "Clear subfields when item cleared."
        ID_REC_GOTO           "Go to a specific record number or
specific ID."
END

```

```

#ifndef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
//
//  Generated from the TEXTINCLUDE 3 resource.
//
#include "res\PTDinp.rc2" // non-App Studio edited resources

#include "afxres.rc"           // Standard components
#include "afxprint.rc"         // printing/print preview resources
#include "afxdb.rc"           // Database resources

/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
#endif // not APSTUDIO_INVOKED

```

```
# Microsoft Visual C++ generated build script - Do not modify
```

```

PROJ = PTDINP
DEBUG = 0
PROGTYPE = 0
CALLER =
ARGS =
DLLS =
D_RCDEFINES = /d_DEBUG
R_RCDEFINES = /dnDEBUG
ORIGIN = MSVC
ORIGIN_VER = 1.00
PROJPATH = C:\DDD\AD97-1\PTDINP\
USEMFC = 0
CC = cl
CPP = cl
CXX = cl
CCREATEPCHFIAG =
CPPCREATEPCHFLAG = /YcSTDAFX.H
CUSEPCHFLAG =
CPPUSEPCHFLAG = /YuSTDAFX.H
FIRSTC =
FIRSTCPP = STDAFX.CPP
RC = rc
CFLAGS_D = WEXE = nologo /G2 /W3 /V /AL /Od /D "_AFXDLL" /D "_DEBUG" /FR
/GA /GEf
CFLAGS_R = WEXE = /nologo /Gs /G3 /W3 /AL /O1 /D "RDEBUG" /D "_AFXDLL" /FR
/GA /GEf
LFLAGS_D = WEXE = NOLOGO /NOD /PACKC:61440 /STACK:10240 /ALIGN:16 /ONERROR:
NOEXE /CO
LIBS_D WRXE = mfc250d oldnames libw llibcew mfcd250d commdlg.lib shell.lib,
LIBS_R WEXE = mfc250 oldnames libw llibcew mfcd250 odbcc commdlg.lib
shell.lib

```

```

RCFLAGS = /nologo /z
RCFLAGS = /nologo /t /k
RUNFLAGS =
DEFFILE = PTDINP.DEF
OBS_EXT =
LIBS_EXT = EVA.LNET_LIB TKSDLL.LIB
!if "$ (DEBUG)" == "1"
CFLAGS = $(CFLAGS_D_WEXE)
LFLAGS = $(LFLAGS_D_WEXE)
LIBS = $(LIBS_D_WIXE)
MAPFILE = nul
RCDEFINES = $(D_RCDEFINES)
!else
CFLAGS = $(CFLAGS_R_WEXE)
LFLAGS = $(LFLAGS_R_WEXE)
LIBS = $(LIBS_R_WEXE)
MAPFILE = nul
RCDEFINES = $(R_RCDEFINES)
!endif
!if (if exist MSVC.BND del MSVC.BND)
!endif
SBRs =          STDAFX.SBR \
                PTDINP.SBR \
                MAINFRM.SBR \
                PTDIDOC.SBR \
                PTDIVW.SBR \
                PTDDLGL.SBR \
                PTDGOTO.SBR

EVA_LNET_DEP =

TKSDLL_DEP =

PTDINP_RCDEP      = c:\ddd\ad97-1\ptdinp\res\ptdinp.ico
                  c:\ddd\ad97-1\ptdinp\res\ptdinp.rc2
STDAFX_DEP = c:\ddd\ad97-1\ptdinp\stdafx.h

PTDINP_DEP = c:\ddd\ad97-1\ptdinp\stdafx.h
              c:\ddd\ad97-1\ptdinp\ptdinp.h
              c:\ddd\ad97-1\ptdinp\ptdidoc.h
              c:\ddd\ad97-1\ptdinp\mainfrm.h
              c:\ddd\ad97-1\ptdinp\ptdivw.h

MAINFM_EP = c:\ddd\ad97-1\ptdinp\stdafx.h
            c:\ddd\ad97-1\ptdinp\ptdinp.h
            c:\ddd\ad97-1\ptdinp\ptdidoc.h
            c:\ddd\ad97-1\ptdinp\mainfrm.h

PTDIDOC_EP = c:\ddd\ad97-1\ptdinp\stdafx.h
             c:\ddd\ad97-1\ptdinp\ptdinp.h
             c:\ddd\ad97-1\ptdinp\ptdidoc.h
             c:\ddd\ad97-1\ptdinp\aa_nets.h

PTDIVW_EP = c:\ddd\ad97-1\ptdinp\stdafx.h
            c:\ddd\ad97-1\ptdinp\ptdinp.h \
            c:\ddd\ad97-1\ptdinp\ptdidoc.h
            c:\ddd\ad97-1\ptdinp\ptdivw.h \
            c:\ddd\ad97-1\ptdinp\ptddlgl.h

PTDDLGL_EP = c:\ddd\ad97-1\ptdinp\stdafx.h

```

```
c:\add\ad97-1\ptdinp\ptdinp.h
c:\ddd\ad97-1\ptdinp\ptdidoc.h
c:\ddd\ad97-1\ptdinp\ptddlgl.h
```

```
all: $(PROJ).EXE $(PRCJ).BSC
```

```
PTDINP.RES: PTDINP.RC $(PTDINP_RCDEP)
$(RC) $(RCFLAGS) $(RCDEFINES) -r PTDINP.RC
```

```
STDAFX.OBJ: STDAFX.CPP $(STDAFX_DEP)
$(CPP) $(CFLAGS) $(CPPCREATEPCHFLAG) /c STDAFX.CPP
```

```
PTDINP.OBJ: PTDINP.CPP $(PTDINP_DEP)
$(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c PTDINP.CPP
```

```
MAINFRM.OBJ: MAINFRM.CPP $(MAINFRM_DEP)
$(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c MAINFRM.CPP
```

```
PTDIDOC.OBJ: PTDIDOC.CPP $(PTDIDOC_DEP)
$(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c PTDIDOC.CPP
```

```
PTDIVW.OBJ: PTDIVW.CPP $(PTDIVW_DEP)
$(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c PTDIVW.CPP
```

```
PTDDLGL1.OBJ: PTDDLGL1.CPP $(PTDDLGL1_DEP)
$(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c PTDDLGL1.CPP
```

```
PTDGOTO.OBJ: PTDGOTO.CPP $(PTDGOTO_DEP)
$(CPP) $(CFLAGS) $(CPPUSEPCHFLAG) /c PTDGOTO.CPP
```

```
$(PROJ).EXE:: PTDINP.RES
```

```
$(PROJ).EXZ:: STDAFX.OBJ PTDINP.OBJ MAINFRM.OBJ PTDIDOC.OBJ
PTDIVW.OBJ PTDDLGL1.OBJ
```

```
PTDGOTO.OBJ $(OBJS_XT) $(DEFFILE)
```

```
echo >KUL @<<$(PROJ).CRF
```

```
STDAFX.OBJ +
```

```
PTDINP.OBJ +
```

```
MAINFRM.OBJ +
```

```
PTDIDOC.OBJ +
```

```
PTDIVW.OBJ +
```

```
PTDDLGL1.OBJ +
```

```
PTDGOTO.OBJ +
```

```
$(OBJS_EXT)
```

```
$(PROJ).EXE
```

```
$(MAPFILE)
```

```
c:\msvc\lib\+
```

```
c:\msvc\mfc\lib\+
```

```
EVALNET.LIB+
```

```
TKSDL.LIB+
```

```
$(LIBS)
```

```
$(DEFFILE);
```

```
<<
```

```
link $(LFLAGS) @$(PROJ).CRF
```

```
$(RC) $(RESFLAGS) PTDINP.RES $@
```

```
@copy $(PROJ).CRF MSVC.BND
```

```
$(PROJ).EXE:: PTDINP.RES
```

```
if not exist MSVC.BND
```

```
$(RC) $(RESFLAGS) PTDINP.RES $@
```

```
run: $(PROJ).EXE
      $(PROJ) $(RUNFLAGS)
```

```
$(PROJ).BSC: $(SBRs)
      bscmake @<<
/o$@ $(SBRs)
<<
```

```
// PTDidoc.h : interface of the CPTDinpDoc class
//
```

```
////////////////////////////////////
```

```
#ifndef _PTDINPDOc_H_
#define _PTDINPDOc_H_
```

```
#define REC_LENGTH 330L
class CPTDinpDoc : public CDocument
```

```
protected: // create from serialization only
    CPTDinpDoco;
    DECLARE_DYNCREATE(CPTDinpDoc)
```

```
//Attributes public:
public:
```

```
    CString m_LAB_ID;
    CString m_NAME_L;
    CString m_NAME_F;
    CString m_NAME_MI;
    CString m_DATE_OF_DATA_ENTRY;           //time
    double m_PATIENT_AGE;
    CString m_DATE_OF_BIRTH;
    CString m_ETHNIC_ORIGIN_WHITE;
    CString m_ETHNIC_ORIGIN_BLACK;
    CString m_ETHNIC_ORIGIN_ASIAN;
    CString m_ETHNIC_ORIGIN_HISPANIC;
    CString m_ETHNIC_ORIGIN_NATIVE_AMERICAN;
    CString m_ETHNIC_ORIGIN_OTHER;
    CString m_MARITAL_STATUS_SINGLE;
    CString m_MARITAL_STATUS_MARRIED;
    CString m_MARITAL_STATUS_DIVORCED;
    CString m_MARITAL_STATUS_WIDOWED;
    CString m_MARITAL_STATUS_LWP;
    CString m_MARITAL_STATUS_OTHER;
    CString m_ACOG_SYMPTOMS;
    CString m_PATIENT_COMPLAINT_1;
    CString m_PATIENT_COMPLAINT_1_1_3;
    CString m_PATIENT_COMPLAINT_1_10_12;
    CString m_PATIENT_COMPLAINT_1_4_6;
    CString m_PATIENT_COMPLAINT_1_7_9;
    CString m_PATIENT_COMPLAINT_1_GTT12;
    CString_rr_PATIENT_COMPLAINT_1_LT1;
    CString m_VAGINAL_BLEEDING;
    CString m_VAGINAL_BLEEDING_TRACE;
    CString m_VAGINAL_BLEEDING_MEDIUM;
    CString m_VAGINAL_BLEEDING_GROSS;
    CString m_PATIENT_COMPLAINT_6;
    CString m_PATIENT_COMPLAINT_3;
    CString m_PATIENT_COMPLAINT_2;
    CString m_PATIENT_COMPLAINT_5;
    CString m_PATIENT_COMPLAINT_4;
```

```

CString m_EGA_BY_SONO;
CString m_EGA_BY_LMP;
CString m_EGA_AT_SAMPLING;
CString m_0_COMP;
CString m_1_COMP;
CString m_2_COMP;
CString m_3_COMP;
CString m_4_COMP;
CString m_5_COMP;
CString m_6_COMP;
CString m_2_COMP_1;
CString m_2_COMP_2;
CString m_2_COMP_3;
CString r_GRAVITY;
CString r_PARITY;
CString r_ABORTIONS;
CString m_MULTIPLE_GESTATION;
CString r_MULTIPLE_GESTATION_TWINS;
CString m_MULTIPLE_GESTATION_TRIPLETS;
CString m_MULTIPLE_GESTATION_QUADS;
CString m_UTCERV_ABNORMALITY;
CString r_CERVICAL_CERCLAGE;
CString m_GESTATIONAL_DIABETES;
CString m_HYPERTENSIVE_DISORDERS;
CString m_DILITATION_LT1;
CString m_DILITATION_1;
CString m_DILITATION_1_2;
CString m_DILITATION_2;
CString m_DILITATION_2_3;
CString m_DILITATION_3;
CString m_DILITATION_GT3;
CString m_DILITATION_UNKNOWN;
CString m_CERVICAL_CONSISTANCY_FIRM;
CString m_CERVICAL_CONSISTANCY_MOD;
CString m_CERVICAL_CONSISTANCY_SOFT;
CString m_ANTIANTIBIOTICS;
CString m_CORTICOSTEROIDS;
CString m_TOYOLYTICS;
CString m_INSULIN;
CString m_ANTIHYPERTENSIVES;
CString m_MEDICATIONS_NONE;
CString m_MEDICATIONS_UNKNOWN;
CString r_FFN_RESULT;

```

```

char Rec[REC_LENGTH + 16];
char fld[256];
char PathName[128];
long CurRecord;
long NumRecords;
int GotoMode;
CString IDStr;
char tstr[2561];
Ctime tim;

```

```

char NetName[128];
char NetRec[1024];
double m_NetPos1;
double M_NetNeg1;
double m_NetVal1;
double m_7NetPos2;
double m_7NetNeg2;

```

```

double m_NetVal2;
double m_NetPos3;
double m_NetNeg3;
double m_NetVal3;

// Operations
public:

void get_rec( char* pRec);
char* get - fld(char* pRec, int ofs, int len);
CTime& get time fld (char* pRec, int of s, int len)
void put_rec(char* pRec);
void put_fld (char* pRec, CString& dat, int of s, int len)
void put_dbl_fld (char* pRec, double dat, int of s, int len);
void put_net_fld (char* pRec, double dat, int of s, int len) ;
void put_time_fld (char* pRec, CTime& dat, int of s, int len)
void InitializeRec(void); void LoadNets(void);
void FreeNets(void);
void RunNets(long n);
char* time2str( const CTime& tm);
CTime& str2time( CString& str);
void get-file( void);

// Implementation
public:
    virtual ~CPTDinpDoc( );
    virtual void Serialize(CArchive& ar); // overridden for document
i/o
#ifdef _DEBUG
    virtual void AssertValid ( ) const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:
    virtual BOOL OnNewDocument ( );
// Generated message map functions
protected:
   //{{AFX_MSG(CPTDinpDoc)
    afx_msg_void OnRecFirst ( );
    afx_msg_void OnRecLast ( );
    afx_msg_void OnRecNext ( );
    afx_msg_void OnRecPrev ( );
    afx_msg_void OnFileOpen ( );
    afx_msg_void OnBldNetFile ( );
    afx_msg_void OnRecGoto ( );
    afx_msg_void OnFileMruFile1 ( );
    afx_msg_void OnFileMruFile2 ( );
    afx_msg_void OnFileMruFile3 ( );
    afx_msg_void OnFileMruFile4 ( );
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP( )
};

#endif // PTDINPDOC_H_

```

```

////////////////////////////////////
////////////////////////////////////

```